AWARD NUMBER: W81XWH-15-C-0193

TITLE: Universal Open Power, Communications, and Control for Assistive Devices

PRINCIPAL INVESTIGATOR: Dr. Michael Grissom

RECIPIENT: KCF Technologies
State College, PA 16801

REPORT DATE: October 2017

TYPE OF REPORT: Annual

PREPARED FOR: U.S. Army Medical Research and Materiel Command
Fort Detrick, Maryland 21702-5012

DISTRIBUTION STATEMENT: Approved for public release; distribution is unlimited.

REPORT DOCUMENTATION PAGE	*Form Approved* *OMB No. 0704-0188*

1. REPORT DATE October 2017	**2. REPORT TYPE** Annual	**3. DATES COVERED** 25 Sep 2016 - 24 Sept 2017
4. TITLE AND SUBTITLE Universal Open Power, Communications, and Control for Assistive Devices		**5a. CONTRACT NUMBER** W81XWH-15-C-0193
		5b. GRANT NUMBER
		5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S) Michael Grissom Email: mgrissom@kcftech.com		**5d. PROJECT NUMBER**
		5e. TASK NUMBER
		5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) KCF Technologies, Inc. 336 S. Fraser St, State College, PA, 16801		**8. PERFORMING ORGANIZATION REPORT NUMBER**
9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES) **U.S. Army Medical Research and Materiel Command** **Fort Detrick, Maryland 21702-5012**		**10. SPONSOR/MONITOR'S ACRONYM(S)**
		11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION / AVAILABILITY STATEMENT

Approved for Public Release; Distribution Unlimited

13. SUPPLEMENTARY NOTES

14. ABSTRACT

To move towards the advanced, powered, interoperable ideal, the next generation of prosthetics must be linked for energy and data flow as well as mechanically. This report covers the first quarter of the technology development project to create an interoperable prosthetic platform.

15. SUBJECT TERMS

None provided

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON** USAMRMC
a. REPORT Unclassified	**b. ABSTRACT** Unclassified	**c. THIS PAGE** Unclassified	Unclassified	101	**19b. TELEPHONE NUMBER** *(include area code)*

Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std. Z39.18

Table of Contents

1. INTRODUCTION

Research has shown that prosthetic device functional interoperability, actuated prosthetics, and higher bandwidth sensing modalities produces improved outcomes. The TATRC Lower Extremity Gait System (LEGS) program advanced the state of the art in defining prosthetic components that can interoperate by sharing power, data, and control even when made by different vendors, but currently, commercial prosthetic components from different vendors operate independently. To move towards the advanced, powered, interoperable ideal, the next generation of prosthetics must be linked for energy and data flow as well as mechanically. Developing and demonstrating an open source platform for lower extremity prostheses will generate a fertile ecosystem for vendors to interoperate and clinical researchers to innovate. The core elements of this platform are open source communications, a flexible energy configuration, advanced high bandwidth sensing, and high energy density actuation technology. This project will advance the state-of-the-art by addressing the primary technical barriers to achieving this ideal. The approach is to demonstrate the range of technologies that will be required for a range of applications, versus a narrowly focused product development approach developing a single product.

2. KEYWORDS

Prosthetic, interoperability, wireless, high bandwidth, assistive, sensors, foot, knee, ankle, socket, evidence-based medicine

3. ACCOMPLISHMENTS

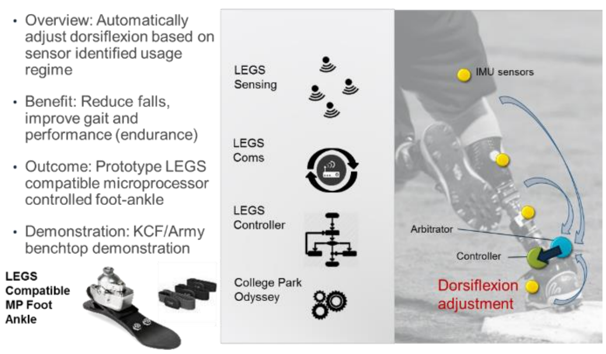
What were the major goals of the project?

The overarching objectives of the LEGS project are to address the following key opportunities:

- Reduce prosthetic user injuries and improve fit by providing intuitive user control, and automatic device environmental awareness
- Create a patient-centric market where the clinicians are not limited by manufacturers' interoperability
- Accelerate prosthetic technology evolution by creating nonexclusive interoperable technology building blocks
- Increase productivity of clinicians by enabling out-of-clinic health & device monitoring

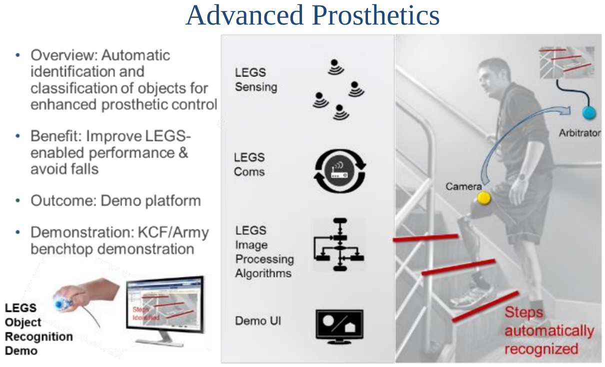
Ultimately this work intends to motivate prosthetics manufacturers to adopt the LEGS concepts and technology. To this end, specific demonstrable outcomes are being developed. The four primary demonstration outcomes are as follows:

Outcome 1: College Park LEGS MP Ankle

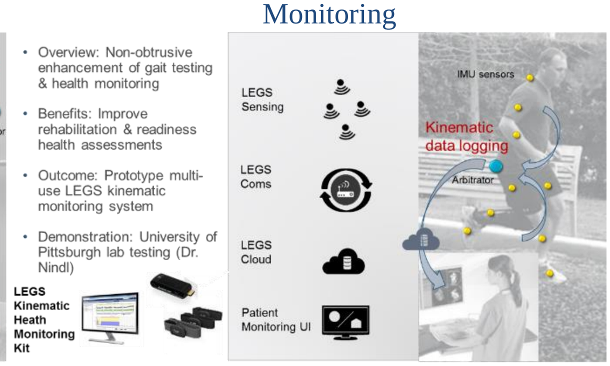


Outcome 2: Socket Health and Usage Monitoring Platform

Outcome 3: Visual Object Recognition for Advanced Prosthetics



Outcome 4: Kinematic Health and Function Monitoring



The specific project goals necessary to make significant progress against these slated outcomes, within the context and limitations of the prosthetics industry, lie in two areas – communications and sensing. Although certain aspects of sensing and communication overlap, they are loosely delineated under this project work effort.

A. Communications Thrust Technical Goals

Aim: Define an interoperable energy storage, transfer, and control platform for assistive devices, and demonstrate key capabilities in hardware demonstrations (annual milestones).

Year 1: Define a wired and wireless, common communication specifications for device interoperability. A bus arbitration scheme and wireless protocol will be specified that enables plug-and-play capabilities. The central objective is to develop a system that will allow for easy adoption by various manufacturers.

Year 2: Define tools to incorporate non-compliant devices into the common communication platform. Components and sensors will be developed to enable clinicians to incorporate devices that do not follow the proposed standards. This includes generating capabilities to externally sensorize conventional devices and modular form factors that fit within current prosthetic components.

Year 3: Define a common external communications platform (clinic or user-based) for data review, social connectivity, and analysis. Bluetooth Low Energy wireless protocol will be added to the wireless infrastructure to support interface with an external portal for supporting communication of high level end-to-end operational data. A software user interface will be developed so that clinicians and users can easily interface with devices.

Communications Thrust Subtasks	**% Done**
C1: Develop sensor platform modules	100 %
C2: Update protocol to handle new sensor functionality	100 %
C3: Develop actuator interface module or translator board	100 %
C4: Prototype and test components and integrate knee-foot platform	100 %
C5: Electrical and mechanical package design	100 %
C6: Develop software algorithm architecture for interoperable dorsiflexion	100 %
C7: Update state machine for external sensors and clinical communications	100 %
C8: Prototype and test components and integrate external sensors platform	100 %
C9: Implement Bluetooth LE stack in wireless radio chip and increase PDCP bandwidth	40 %
C10: Design and develop user interface software	40 %
C11: Iterative mechanical and electrical package design for full technology integration	0 %
C12: Prototype and test components and modules	5 %

B. Sensing Thrust Technical Goals

Aim: Define patient-centric high bandwidth sensors and platform for optimal human-device interoperability, and demonstrate key capabilities in hardware demonstrations (annual milestones).

Year 1: Define a high bandwidth common sensor architecture. The central processing unit and modular sensor units will be defined for the system. Open source technology developed for visual tracking in robotics will be integrated as the baseline technology.

Year 2: Define a common data fusion architecture to combine multiple high bandwidth sensor data streams. An architecture to combine complementary data measurements and operate on the data stream will be implemented. A gait characteristic will be chosen to optimize in demonstration.
Year 3: Define an integrated mechanical and electrical plug-and-play capability for high bandwidth sensor information. A software API and firmware embedded state machines will be implemented to generalize the platform for future applications. The primary objective is to design measurement flexibility into the system.

Sensing Thrust Subtasks	**%Done**
S1: Develop modular high bandwidth sensors	100 %
S2: Design circuitry for high bandwidth open architecture sensor platform	100 %
S3: Integrate data from multiple sensors into microprocessor	100 %
S4: Prototype and test sensors and sensor platform for demonstration	100 %
S5: Define high bandwidth sensor and platform design specifications	100 %
S6: Sensor platform detail design component integration	100 %
S7: Software/firmware implementation for gait modification	100 %
S8: Prototype and test integrated sensors for technology demonstration	100 %
S9: Design open component electrical interface for sensor platform	0 %
S10: Design open component mechanical interface for sensor platform	5 %
S11: Integrate sensors and platform with multiple vendor devices	10 %
S12: Prototype and test smart sensor system for technology demonstration	0 %

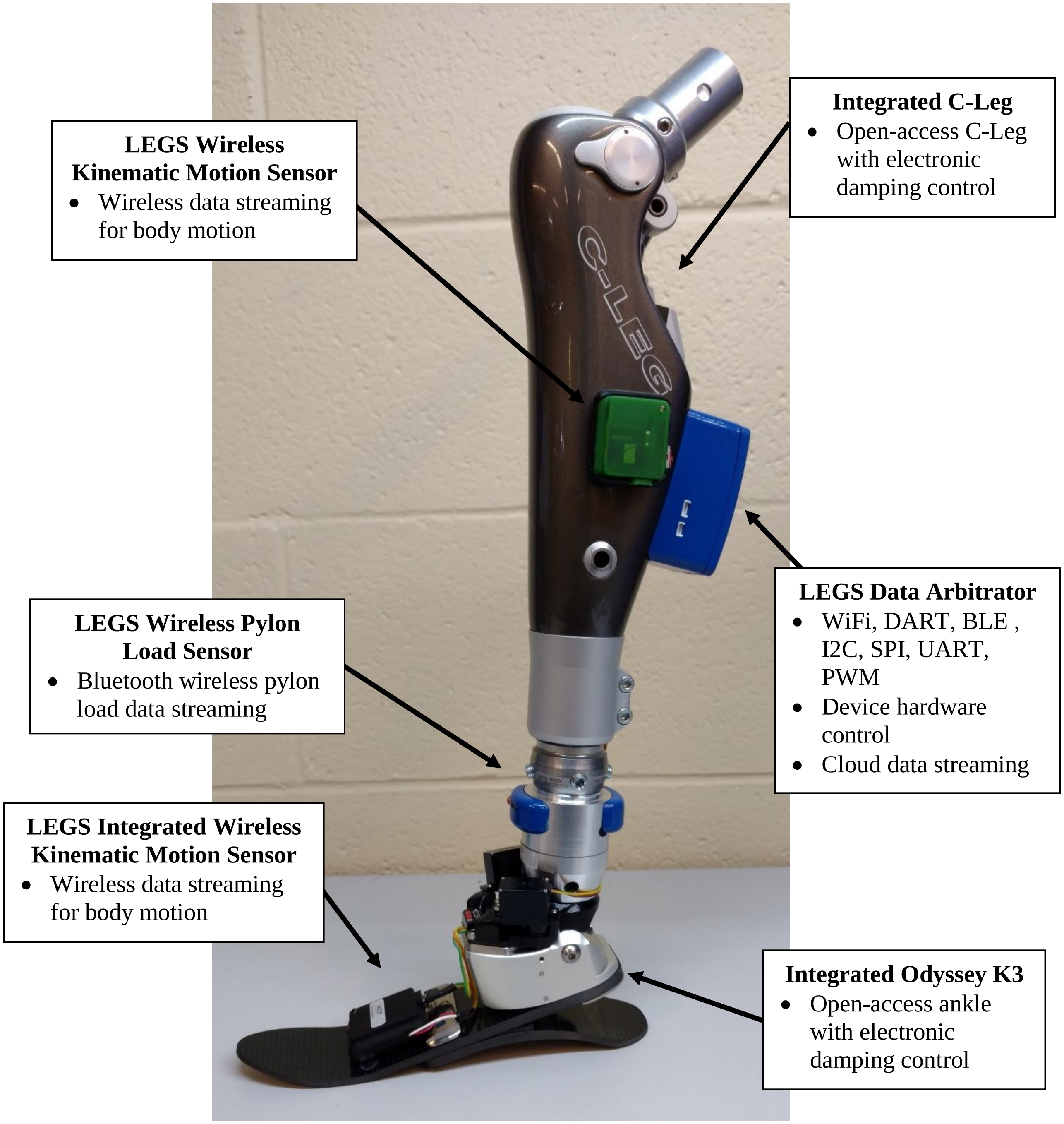
What was accomplished under these goals?

A. Overall Progress Summary

BASE PROJECT MILESTONE OVERVIEW			
	YEAR 1	**YEAR 2**	**YEAR 3**
Communications (KCF, LTI)	☑ Multi-vendor knee-foot interoperability with internal vendor sensor control of second vendor actuation	☑ Multi-vendor knee-foot interoperability with external vendor sensors: integration of knee and foot dorsiflexion angle for level ground walking improvement	Smart Pylon device with multi-vendor knee-foot interoperability via parallel bus-based wired technology and wireless Bluetooth LE. Integration of a suite of existing and research- based input sensors and actuators to the communication backbone
Sensing (KCF, PU)	☑ Streaming high bandwidth sensor data (EMG, RGB-D, IMU, Fiber Bragg) controlling an assistive device (knee or foot) actuation	☑ Multi-vendor knee-foot interoperability with streaming fused high bandwidth sensor data for gait optimization	Smart device with high bandwidth streaming sensor data plug-and-play capability for multi-vendor open control architecture

The progress completed during the second year of the LEGS project has fulfilled the planned technical advancement and maturity necessary to complete the overall project objectives. Significant progress was made in developing the open interoperable prosthetic system; major milestones include the following achievements:

- <u>Version 2 LEGS Data Arbitrator completed</u>
    - Includes wireless sensor and prosthetic hardware data streaming to cloud
    - Supports WiFI, Bluetooth Low Energy, and DART wireless I/O protocols
    - Supports analog, I2C, SPI, UART, and PWM wired I/O protocols
    - Features swappable, rechargeable battery modules
    - Support for generic sensor inputs and expandability
- <u>C-Leg integrated</u> with open access to digital control of knee damping rates
- <u>Odyssey K3 foot integrated</u> with open access to digital control of ankle damping rates
- <u>LEGS User Interface</u> created and includes the following features:
    - Live sensor data streaming and data recording/storage
    - Derived (calculated) body and prosthetic position data based on sensor data inputs
    - 3D patient avatar, showing real time and recorded prosthetic and body gait motion
    - Sensor status widget, showing connectivity and battery status
    - Sensor body/prosthetic placement widget
    - User and Clinician patient alerts, alarms, and text messages, custom configurable for falls, angle exceedances, deleterious motions and behaviors, etc
- <u>Multiple IMU body motion sensors manufactured</u>, including wearable hardware, and integration with prosthetic hardware
- <u>Version 1 open-access wireless Bluetooth sensor design, manufactured, and integrated</u> with LEGS system
- <u>Coordinated control of C-Leg knee and Odyssey ankle implemented and tested</u>
- <u>Coordinate control algorithm architecture map and input matrices completed</u> for Scenario 1. Standing on Uneven Slopes, work started on Scenario 2. Stair Descent
- <u>Concept completed for proposed PDCP replacement</u> – an Open, Universal, Prosthetic Software Translator, developed in partnership with LTI and CoApt
- <u>High bandwidth optical sensor and stair detection system completed</u> in partnership with Purdue University
- <u>System and component level testing completed</u>
- <u>Support built for integration of KCF’s Smart Socket</u>


LEGS Wireless Kinematic Motion Sensor
- Wireless data streaming for body motion

Integrated C-Leg
- Open-access C-Leg with electronic damping control

C-LEG

LEGS Data Arbitrator
- WiFi, DART, BLE , I2C, SPI, UART, PWM
- Device hardware control
- Cloud data streaming

LEGS Wireless Pylon Load Sensor
- Bluetooth wireless pylon load data streaming

LEGS Integrated Wireless Kinematic Motion Sensor
- Wireless data streaming for body motion

Integrated Odyssey K3
- Open-access ankle with electronic damping control

LEGS User Interface, showing 3D patient motion avatar, sensor placement and status widget, activity statics, patient alerts and alarms, and configurable data displays showing knee angle, etc, and raw sensor data

Year 3 Plan:

Task planning for Year 3 is in progress. KCF has made significant progress in Year 2 on some Year 3 objectives, including implementation of BLE protocol; development of the software UI; integration of existing prosthetic protocols; and concept development of open protocol translation in place of PDCP.

Based on the SOW and the work completed so far, Year 3 LEGS tasks will focus primarily on the following:

1. Development of the open software translation system for prosthetic devices in place of the defunct PDCP protocol
2. Coordinate control algorithm coding and implementation
3. Further development of the LEGS user interface
4. Further development of open electrical and mechanical interfacing for the LEGS sensor/hardware ecosystem
5. Integration of additional microprocessor prosthetic devices (e.g. Endolite Elan foot, Orthocare Europa + System)
6. Partnerships with existing prosthetic manufacturers to integrate their devices with the LEGS open system (Endolite, Freedom, etc), and transition the LEGS system to market

		Oct. 2016	Nov. 2016	Dec. 2016	Jan. 2017	Feb. 2017	Mar. 2017	Apr. 2017	May. 2017	Jun. 2017	Jul. 2017	Aug. 2017	Sep. 2017	Oct. 2017	Nov. 2017
		Year 2													
	Report #:	14	15	16	17	18	19	20	21	22	23	24	25	26	27
Communications													◆		
	C5: Electrical and mechanical package design												◆		
	Arbitrator package design	◆													
	V1 Arbitrator development		◆												
	Coordinated knee/ankle hardware design												◆		
	V2 Arbitrator development									◆					
	Ankle hardware electro-mechanical control development										◆				
	V2 Arbitrator hardware integration											◆			
	Arbitrator BLE, UWB integration												◆		
	C6: Develop software algorithm architecture for interoperable dorsiflexion												◆		
	Build software support for conditional hardware triggering			◆											
	Develop conditional statements for knee coordinated control					◆									
	Develop conditional algorithms for knee/ankle coordinated control												◆		
	Finalize coordinated control of knee/ankle hardware for Scenario 1												◆		
	C7: Update state machine for external sensors and clinical communications												◆		
	Build software support (arbitrator) for I/O data streaming				◆										
	V1 LEGS UI: Build software support for clinical streaming of data														
	Build analysis tools for clinical use of sensor data stream								◆						
	V1 LEGS UI: Clinical presentation of data stream and prosthetic health and status												◆		
	Add multi-user, multi prosthetic oversight to V2 LEGS UI														
	C8: Prototype and test components and integrate external sensors platform												◆		
	Prototype coordinated knee/ankle												◆		
	Add additional sensor support for socket monitoring to arbitrator/communications platform												◆		
	Controlled ankle integration to demo platform												◆		
	High bandwidth sensor integration												◆		
Sensors													◆		
	S5: Define high bandwidth sensor and platform design specification												◆		
	Test high bandwidth sensor for optical object recognition		◆												
	Develop and manufacture high bandwidth optical detection sensor											◆			
	S6: Sensor platform detail design component integration												◆		
	Develop and integrate IMU sensor into LEGS environment				◆										
	Integrate multiple kinematic sensors into LEGS environment						◆								
	Develop pylon load and moment sensor										◆				
	Develop and integrate BLE sensor into LEGS environment												◆		
	Transition socket sensors (SBIR) with LEGS environment														
	S7: Software/firmware implementation for gait modification												◆		
	IMU sensor firmware development			◆											
	Develop data compression for high density sensor data streams					◆									
	Analyze Scenario requirements to define sensor measurement details							◆							
	Develop preliminary gait control algorithms										◆				
	Finalized development of Scenario 1 gate control algorithms												◆		
	S8: Prototype and test integrated sensors for technology demonstration												◆		
	Collect and analyze preliminary sensor data input to LEGS arbitrator					◆									
	Prototype & demonstrate initial hardware control via sensor input					◆									
	Mature sensor suite input to LEGS environment												◆		
	Benchtop test coordinated knee/ankle control										◆				
	Develop and implement plug & play functionality to LEGS environment												◆		
	Demonstrate V2 arbitrator, V2 LEGS UI, and integrated knee/ankle control for Scenario 1												◆		
Milestones															
C	Multi-vendor knee-foot interoperability with external vendor sensors: integration of knee and foot dorsiflexion angle for level ground walking improvement													▲	
S	Multi-vendor knee-foot interoperability with streaming fused high bandwidth sensor data for gait optimization													▲	

B. Individual Task Progress

Communications Thrust Aim

Goal: Define and develop wired and wireless common communication specifications for device interoperability. A bus arbitration scheme and wireless protocol will be developed that enables plug-and-play capabilities. The central objective is to develop a system that will allow for easy adoption by various manufacturers.

C5: Electrical and mechanical package design

- All C5 subtasks for Year 2 completed
- V1 Edison-based Data Arbitrator completed: first prototype unit assembled, including electronics, overmolding and packaging, C-Leg mounts, and analog and digital control I/Os
- Transition made from Intel Edison based V1 Data Arbitrator to V2 Raspberry Pi Zero W based Arbitrator due to Intel's discontinuation of the Edison platform; code ported to Pi unit and made compatible with the new platform
- V2 Pi-based Data Arbitrator electrical, mechanical, and communications packages designed, fabricated and completed. V2 Data Arbitrator functionality includes:
    - Analog and digital GPIOs for prosthetic hardware control
    - Bluetooth Low Energy, WiFi, DART wireless protocols
    - 2000mAh swappable LiPo battery modules
    - Shared power distribution
    - Low voltage soft shutdown & LiPo battery protection
    - On/off switch with LED indicator
    - Wireless firmware update capability
- College Park Odyssey K3 ankle received and successfully integrated with the LEGS control system. Odyssey foot was integrated with microprocessor control hardware on the plantar & dorsiflexion damping control valves using digital servo motors
- Coordinated knee/ankle hardware design completed – C-Leg and Odyssey foot integrated with open platform control of knee and ankle flexion/extension damping
- V1 wireless pylon load sensor completed: load sensor includes Bluetooth Low Energy wireless, rechargeable LiPo battery, and compact profile. A V2 load sensor is under development and marketing may be pursued
- Open Bluetooth Low Energy protocol integrated with the LEGS Arbitrator and sensor system

Task C5 Detailed Description

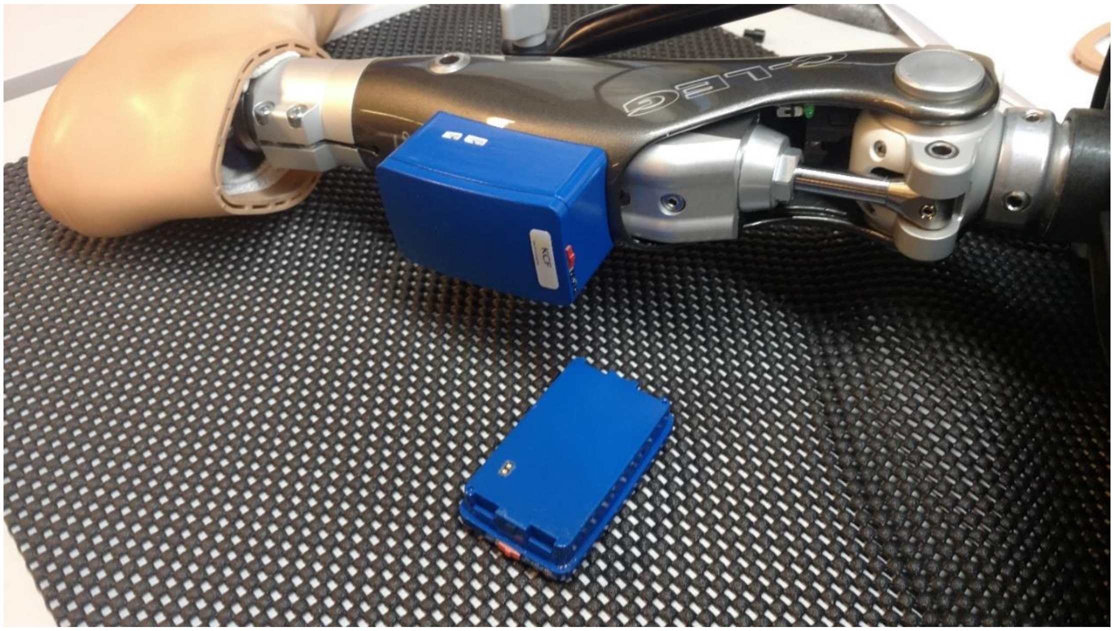
Work under Task C5 in Year 2 focused on assembly, programming, and development of the prototype V1 Intel Edison-based Data Arbitrator and V2 Raspberry Pi Zero W-based Data Arbitrator, mechanical and electrical integration of the C-Leg knee and Odyssey K3 foot, and development and fabrication of the wireless IMU and pylon load sensors. C-Leg knee and Odyssey ankle control has been integrated with the LEGS arbitrator system, allowing the open prototype LEGS control system to adjust both prosthetic devices in real time.

The data Arbitrator and packaging is complete, with a mountable housing and modular swappable battery system. The Arbitrator protocols consist of WiFi, DART wireless, Bluetooth, ADC inputs, I2C, SPI, and UART.

BLE support was added using the open source BlueZ package for Raspbian and Python interface, and demonstrated via the BLE wireless pylon load sensor developed by KCF. Like other wireless sensor inputs to the LEGS system, BLE data is received by the Arbitrator and sent to the LEGS

cloud via websocket. A V2 load cell is currently being manufactured to reduce size and mature the wireless pylon load cell design for potential market transition.

Data Arbitrator Development



LEGS Data Arbitrator and modular battery packs

The V1 and V2 Arbitrator electronics and packaging were successfully completed and tested for conditional control of the C-Leg knee and Odyssey foot this year. Following Intel's announced discontinuation of the Edison platform, work shifted to successfully incorporate the Raspberry Pi Zero W as the main Arbitrator platform.

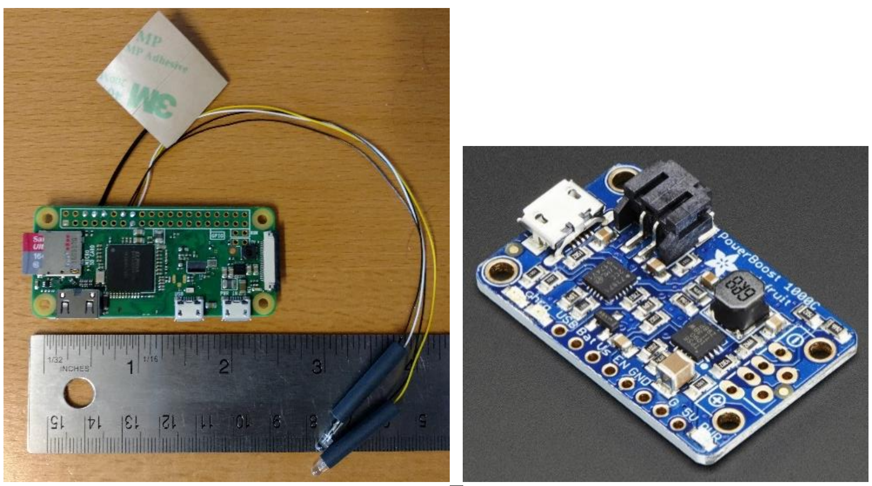
The Raspberry has largely the same functionality of the Intel Edison module, with the added benefit of being smaller, running Linux natively, and incorporating Wifi, Bluetooth, GPIOs (PWM, UART, analog, etc) on a single board. The Edison required additional modules to be stacked to add functionality (such as UART interface) which while convenient, increased power consumption and package size. Transition to the Raspberry Pi based arbitrator presented only about a week-and-a-half setback, as work was already in progress on the V2 arbitrator which the Raspberry Pi slotted into smoothly.

The change to Raspberry Pi Zero W has proven to be a benefit for the project as it is much easier to interface with and program, better supported by Raspberry and the open source community, utilizes what has become a standard 40 pin GPIO interface, allows the use of all legacy Raspberry Pi interface boards, is smaller in size, and builds upon existing legacy Raspberry boards. In addition, the Pi has an integrated SD card slot for expandable memory, 3.3V output built in, graphics chip and camera input, and SPI and I2C interfaces. Raspberry has traditionally followed a path of compatible upgrades versus strict obsolescence, which ensures that any replacement modules in the future will preserve or enhance the functionality of the existing module while remaining backwards compatible.

<ins>Raspberry Pi Zero W</ins>

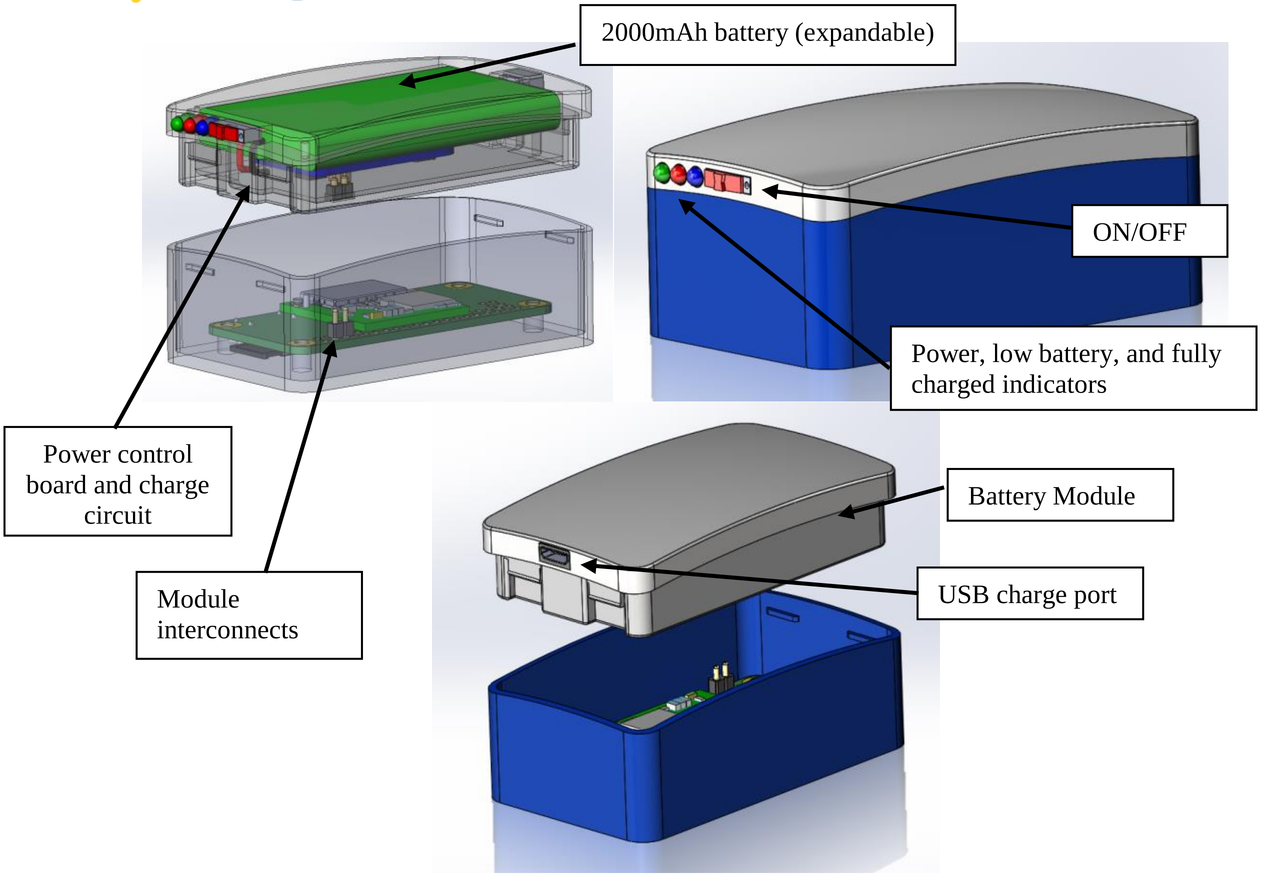
- 1GHz, single-core CPU
- 512MB RAM
- Mini-HDMI port
- Micro-USB On-The-Go port
- Micro-USB power
- HAT-compatible 40-pin header
- Composite video and reset headers
- CSI camera connector
- 802.11n wireless LAN
- Bluetooth 4.0

To provide LiPo battery power to the Pi arbitrator and the shared power LEGS components, a LiPo safe battery control board was sourced, the Adafruit PowerBoost 1000C. This board provides 5V 1A output to the Pi and other connected devices, a LiPo safe charge circuit, and low voltage soft shutdown of the Pi as the battery discharges.



Left: Arbitrator V2 to replace the discontinued Intel Edison: Raspberry Pi Zero W with KCF Dart wireless interface on the bottom, with antenna. Right: PowerBoost 1000C for providing 5V battery power to the Pi Arbitrator, safe LiPo charging, and low power soft shutdown

KCF designed and produced an arbitrator housing with modular battery for the V2 Pi based arbitrator. The design consists of a simple housing for the arbitrator electronics and I/Os along with a swappable battery module. The battery component is a standalone unit with the battery and charge circuit, LED indicators (for power, low battery, and battery charged), on/off switch, and USB charge port. Connection between the Pi and the battery module is achieved with spring pin connectors. This will allow the user to swap batteries as needed and charge the drained battery via a USB port or wall plug while the prosthetic is in use with a second battery.



V2 Pi based arbitrator housing, showing the modular battery and components

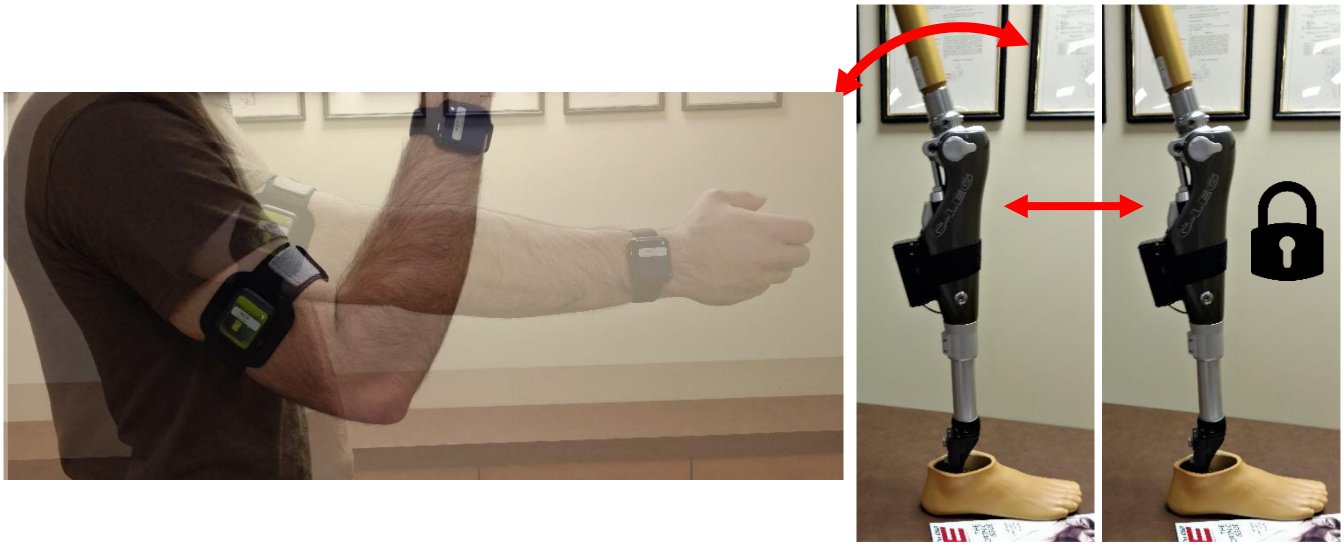
The small size of the Raspberry Pi made a significant amount of space available in the arbitrator packaging over the Edison based arbitrator. This allowed us to increase the battery capacity from 1100mAh to 2000mAh, which will be critical for shared power distribution. This design also lends itself to fitting higher capacity batteries with a taller lid if needed, similar to extended laptop or cell phone batteries.

A customized data support package was developed for the LEGS wireless board; it utilizes the DAART wireless protocol for network communication, while customizing the collection and transmission of the IMU data. This package uses a single SPI bus to communicate with the IMU sensor; IMU data is collected at 50hz and processed by the DCM (Direction Cosine Matrix) algorithm. The resulting orientation vectors are transmitted over the wireless link at 10 Hz. Calibration capabilities are an important aspect in guaranteeing accurate results; these capabilities are built directly into the wireless board support package using customized commands and responses to and from the remote host.

To ensure reliable communication while the sensors are transmitting data, enhancements to the DAART wireless protocol were added to disable network agility capabilities. Therefore, once the sensor is connected to a network and put into a data collection mode, it will continuously transmit to that network until otherwise commanded to.

To demonstrate Arbitrator data computation and conditional outputs, a test was setup where the arbitrator uses the orientation data of two IMU nodes to calculate the angle between those nodes. This represents the angle of the joint between the nodes, e.g. the knee, or for this demonstration,

the elbow. The angle is then used to toggle the outputs which control the knee of the prosthetic leg. An angle of 45 degrees was arbitrarily chosen as the toggle point, with a hysteresis of 5 degrees. Also, a 100ms blanking interval was chosen to prevent quickly fluctuating between states.

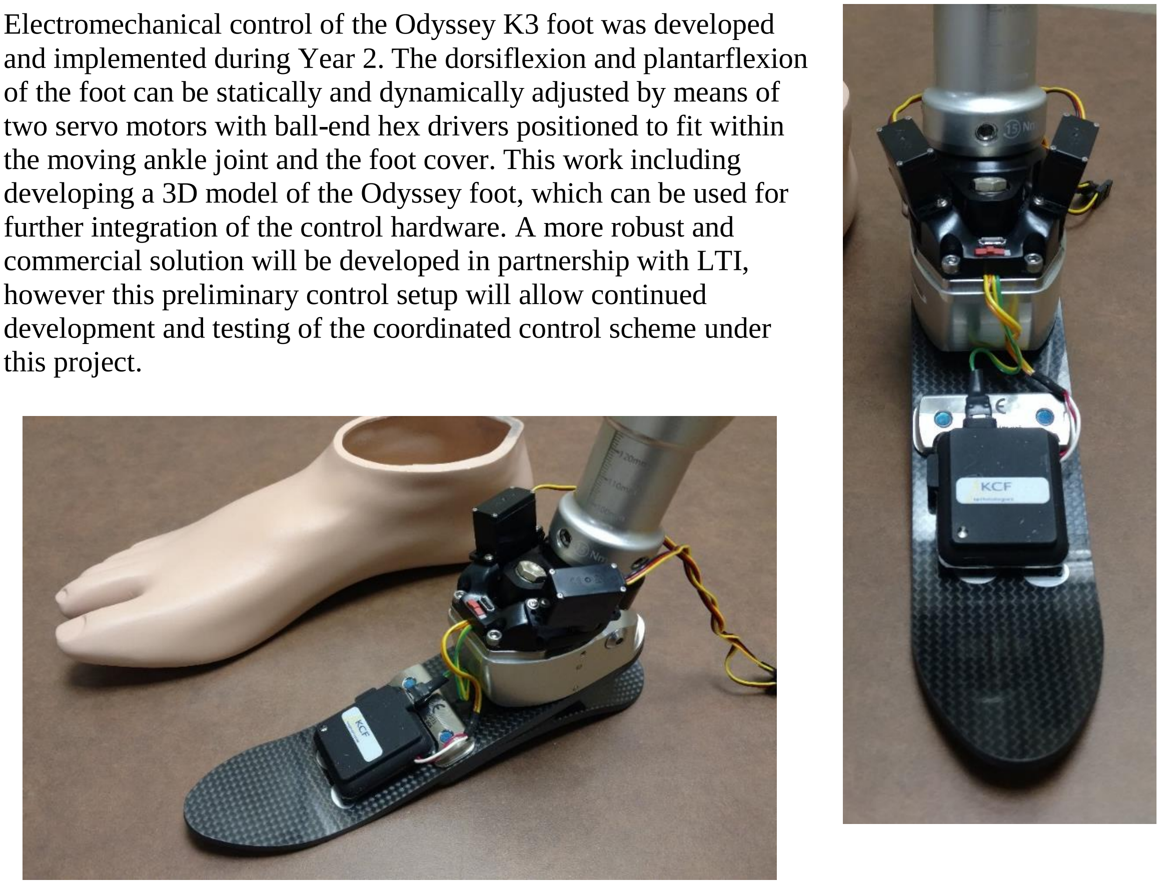


LEGS biometric control of prosthetic hardware – for the first demonstration, two IMUs are worn on the arm. When the arm is bent at the elbow, the arbitrator detects the orientation and commands the modified C-Leg to change the damping state of the knee. Arm movement is used in this demonstration for ease of testing and demonstration, however the same concept applies to coordination between foot, ankle, knee, and thigh mounted kinematic monitoring

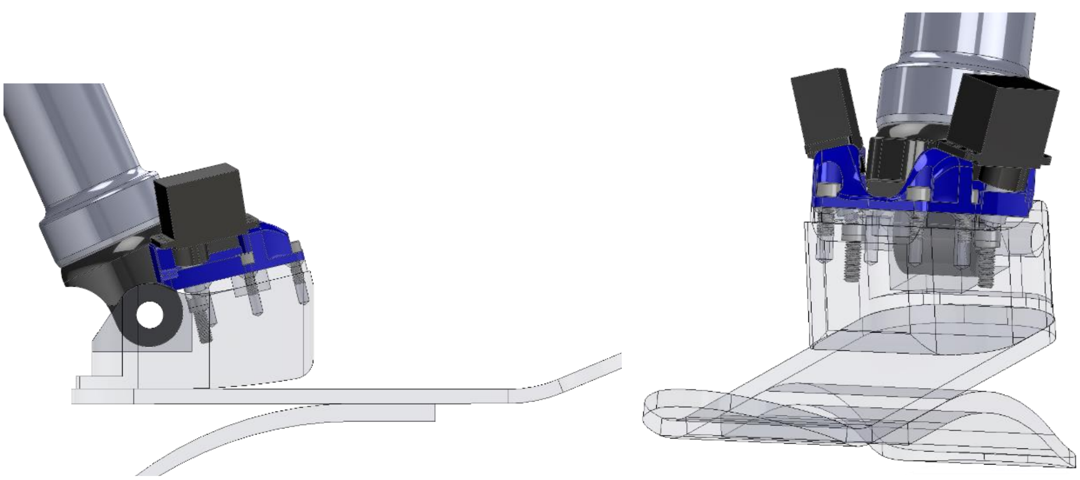
The 0-3V output from the arbitrator is sent to an H-bridge mounted inside the pylon of the C-Leg; the H-bridge is hardwired to the damping adjustment motor in the C-Leg damping unit, adjusting the damper to full on or full off. In this demonstration, the knee can be either locked or unlocked, however with the addition of the encoder various damping states and control can be achieved. KCF is pursuing I2C control of the C-Leg to replace the current analog 0-3V

College Park Odyssey K3 Foot Integration

Electromechanical control of the Odyssey K3 foot was developed and implemented during Year 2. The dorsiflexion and plantarflexion of the foot can be statically and dynamically adjusted by means of two servo motors with ball-end hex drivers positioned to fit within the moving ankle joint and the foot cover. This work including developing a 3D model of the Odyssey foot, which can be used for further integration of the control hardware. A more robust and commercial solution will be developed in partnership with LTI, however this preliminary control setup will allow continued development and testing of the coordinated control scheme under this project.

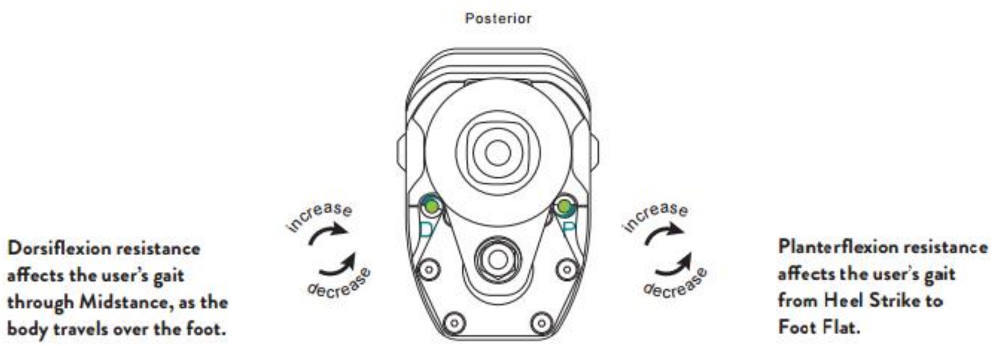


K3 Odyssey foot updated with servo motors to control dorsiflexion and plantarflexion resistance. In addition, the dedicated foot-mounted IMU is shown with remote charge port and ON/OFF switch integrated with the servo mount block

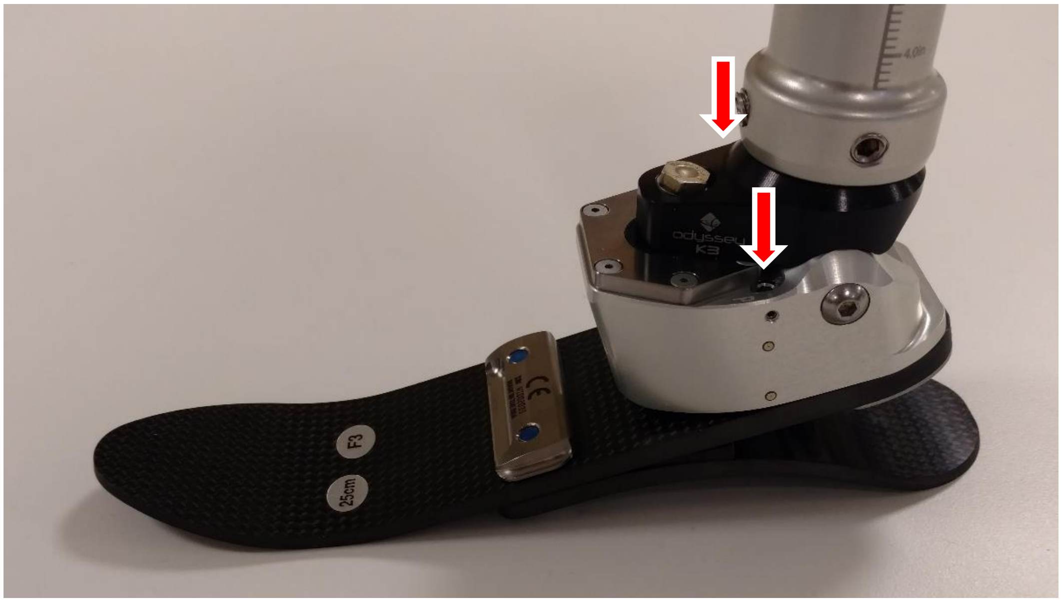


3D model of the Odyssey foot showing the integration of the servo motors with the plantar and dorsiflexion adjustment screws

The foot has two main adjustments, plantarflexion and dorsiflexion resistance, effecting 12deg of plantar/dorsiflexion movement. These valves are typically manually adjusted and set by means of a hex wrench in a clinical environment. The servos utilized will allow high speed, encoded adjustment to set the correct resistance for the detected kinematic state and algorithmically determined resistance.



Resistance adjustments of the Odyssey foot



College Park Odyssey K3 foot, with manually adjustable plantarflexion and dorsiflexion (red arrows) – KCF will fit this foot with automatically and dynamically adjustable flexion and a fixed IMU

Further development of the electromechanically controlled Odyssey K3 foot has been handed off to LTI, who have improved the mounting structure of the servo motors which control the damping adjustment valves. A smaller and lighter servo capable of greater range of rotation was identified (Futaba S3154) and incorporated onto the Odyssey foot. KCF and LTI will continue developing the interface and power sharing between the prosthetic hardware components (foot and knee), while LTI pursues commercial development of the microprocessor controlled K3 foot.

KCF and LTI have received feedback from prosthetists indicating that the Endolite Elan foot is the most popular microprocessor controlled foot on the market. During Year 3, KCF proposes to integrate the Elan foot into the LEGS system in addition to the Odyssey foot; this will demonstrate an additional off-the-shelf prosthetic device on the LEGS platform. KCF also intends to work with Endolite regarding their communications protocol with the aim to make them a partner on the open translator software replacing PDCP.

C6: Develop software algorithm architecture for interoperable dorsiflexion

- All C6 subtasks for Year 2 completed
- Implemented and demonstrated preliminary biometric sensor input conditional control algorithm and outputs from the Arbitrator for knee control
- Two key demonstration scenarios were identified for development and demonstration of coordinated control: 1. Standing on uneven slopes; 2. Walking down stairs
- The standing control input matrix has been completed in partnership with LTI:
    - Required sensor inputs are confirmed to include three, possibly four inputs: pylon load; foot acceleration; knee angle acceleration; and potentially knee/ankle moments
    - KCF has completed sensors for three of the required inputs with additional sensor data available. Knee & ankle angles and rates will be covered via IMU, knee and ankle loads via the load sensor
- Coding the coordinated control algorithms will commence in Year 3 under Task C11 – “Iterative mechanical and electrical package design for full technology integration”
- Algorithm architecture for leading into and out of standing was completed. This matrix and diagram will inform the development of the coordinated control algorithms in Year 3
- Measuring hydraulic fluid pressure in the Odyssey foot is being investigated to provide ankle torque measurement

Task C6 Detailed Description

All C6 tasks for Year 2 have been completed. Sensor-based conditional triggering of prosthetic hardware was demonstrated this year, with preliminary conditional coding statements developed. Two use-case scenarios were identified to demonstrate LEGS interoperability: Scenario 1 -

standing on uneven slopes, and Scenario 2 – walking down stairs. Scenario 1 investigation began within Year 2, resulting in a required sensors list and algorithm matrix control strategy.

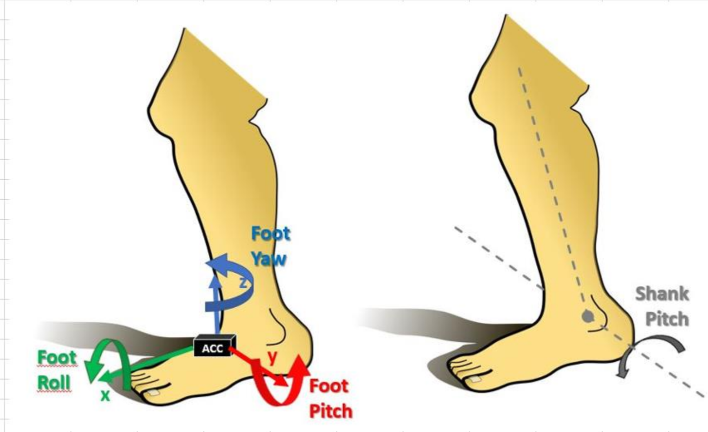
Coding the algorithms for interoperable coordinated control will begin in Year 3.

The tasks to implement coordinated control were developed and completed in partnership with LTI:

- Year 2 - Defined scenarios of interest for demonstration of coordinated ankle/knee control:
    - Scenario 1 (Simple) – Standing on slopes: This scenario was evaluated first and seeks to resolve an extant issue where current prosthetic feet/ankle/knee systems do not lock appropriately when the user is standing on a slope (i.e. do not lock the ankle at the proper angle to match the slope); this results in the user placing most of their weight on their able leg, leading to instability, poor posture, and potential injury. By coordinating the angle of the ankle and the locked states of the ankle and knee, a more stable and firmly planted posture is sought when standing on sloped surfaces.
    - Scenario 2 (Complex) – Stair descent: Stair descent/ascent has traditionally been the user scenario requiring the greatest development in prosthetic systems. LTI has identified that stair **ascent** is more addressable with *actuated* ankles, whereas **descent** is addressable with *passive* ankles. Using their Odyssey passively controlled ankle as a starting point, this scenario will seek to coordinate the passive function of a microprocessor controlled Odyssey ankle with kinematic motion to improve stair descent.
- Year 2 - Defined sensor inputs and data formats required to identify the chosen biometric scenarios and provide actionable sensor input to the system to achieve coordinated control. This task included identifying the biometric states that are unique to the chosen scenarios and determining what measurements are needed and where they must be taken (accel, gyro, magnetometer, etc, measured at foot, ankle, able thigh, etc)
- Year 2 - Mapped the algorithmic approach to the coordinated control logic between the knee and ankle for the two scenarios
- Year 3 - Write algorithms with coordinated control logic between knee and ankle, including digital or analog outputs for controlling the prosthetic hardware (Boolean or proportional signal output based on gait cycle and scenario)

The full standing control input matrix has been completed and is included in Appendix C (sample shown below). The sensor inputs needed to control the standing scenario include: load on the prosthesis, accelerations of the foot, knee angle, and possibly knee and/or ankle moments. The IMU sensors will provide foot, lower leg, upper leg, and opposing leg orientation, acceleration, and angle, allowing the Arbitrator to calculate foot and knee angles and rate of change. The pylon load cell will provide load data on the prosthesis.

Assumptions:	Ankle +	Dorsiflexion
	Ankle -	Plantarflexion
	Knee +	Flexion
	Knee -	Extension
	Forces and Moments are reported as external	
	x	direction of progression (+ forward)
	y	medial / lateral direction (postive left)
	z	vertical direction (positive up)
Ankle	roll	rotation about x axis
Ankle	pitch	rotation about y axis (positive up)
Ankle	yaw	rotation about z axis
Shank	pitch	rotaton about the y axis
		(postive is leaning towards forward progression)
KEY:	PTO	Prosthetic Toe-Off
	PIC	Prosthetic Initial Contact
	PFF	Prosthetic Foot Flat
	ITO	Intact Toe Off
	IIC	Intact Initial Contact
	IFF	Intact Foot Flat
LEGEND:		potential inputs for transition into/out of standing



Control Input Matrix Nomenclature

Standing - Pros Lead In & Out

Note: values given for Level Walking. Some variables would change depending on slope (e.g. M_A, M_K, Θ_A, ω_A, & $pitch_{foot}$)

Sensor	PIC	PIC-PFF	PFF	PFF-ITO	ITO	ITO-IIC	IIC	IIC-IFF	IFF	Standing		Stand - PF(	PFO	
	Event	Transition	Event	Transition	Event	Transition	Event	Transition	Event	wait = 2sec		Transition	Event	
Fv	+ (but small)	incr.	+ (but small)	incr.	BW	incr., decr, incr.	~BW	decr.	1/2 BW*	constant		decr.	0	
M_A	- (but small)	decr. (plantar)	-	incr (dorsi)	0	incr.	max +	decr.	~0*	constant		constant	0	
M_K	- (but small)**	decr, then incr.	-	incr.	~0	incr, then decr	~0	constant	~0*	constant		constant	small or 0	
Θ_A	+	decr. (plantar)	-	incr (dorsi)	- (but small)	incr (dorsi)	~0	constant	~0*	constant		constant	0	
ω_A	- (but small)	decr, then incr.	0	incr.	+	incr, then decr	~0	constant	~0*	constant		constant	0	
Θ_K	~0	constant	0	constant	0	constant	0	constant	~0*	constant		constant	0	
ω_K	~0	constant	0	constant	0	constant	0	constant	~0*	constant		constant	0	
ACCx	~0	constant	0	constant	0	constant	0	constant	0	constant		incr.	+***	
ACCy	~0	constant	0	constant	0	constant	0	constant	0	constant		constant	~0***	
ACCz	- (but small)	decr, then incr.	0	constant	0	constant	0	constant	0	constant		incr.	+***	
$roll_{foot}$	~0	constant	0	constant	0	constant	0	constant	0	constant		constant	0***	
$pitch_{foot}$	+	decr. (plantar)	0	constant	0	constant	0	constant	0	constant		constant	0***	
yaw_{foot}	~0	constant	0	constant	0	constant	0	constant	0	constant		constant	0***	
$pitch_{leg}$	-	incr.	-	incr.	~0	constant	~0	constant	~0	constant		constant	0	
Gait cycle	*0%*		*8%*		*10-12%*		*50%*		*58%*					

*within a small, but reasonable range due to sway and weight shifting

**due to knee being fully extended (0°) at heel strike. If SPKF or uphill slope, then it could be a initial + moment.

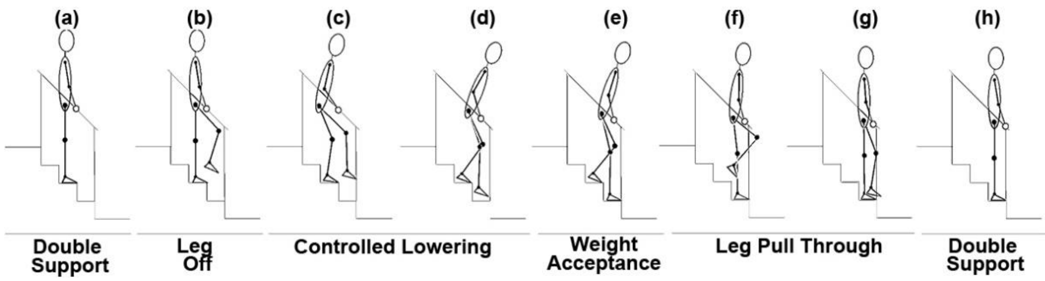
***some small changes may be seen depending on how the foot is off-loaded (e.g. hip hiking, circumduction, etc.)

Subset of the Standing Control Input Matrix, showing expected forces, moments, and angles during lead in and lead out

In order to derive the requirements for coordinate control during stair descent (one of the more difficult and potentially painful common tasks for patients), our research partner LTI (College Park) has evaluated and identified the requirements to successfully implement a coordinated C-Leg/ankle prosthetic strategy; in order to achieve ankle/knee coordination the follow requirements should be met:

- The knee joint should have individually controllable extension and flexion resistance, coordinated with the ankle plantar and dorsiflexion.
- Able-bodied individuals plantar flex their foot by ~20-30 degrees before making contact with the next stair. By coordinating plantar flexion with leg swing during descent, then increasing dorsiflexion resistance after toe contact is made, a smoother, safer, and more natural lowering and weight transition can be achieved while weight is transferred to the prosthetic limb. (See steps b-e below)
- Ankle dorsiflexion angles may need to be greater for a prosthetic ankle (20-30 degrees) when compared to able-bodied ankles to allow controlled lowering of the sound limb to the next step. This is because unlike able-bodied ankles/feet, there is no articulation of the metatarsal heads (ball of the foot). (See steps d-e below)

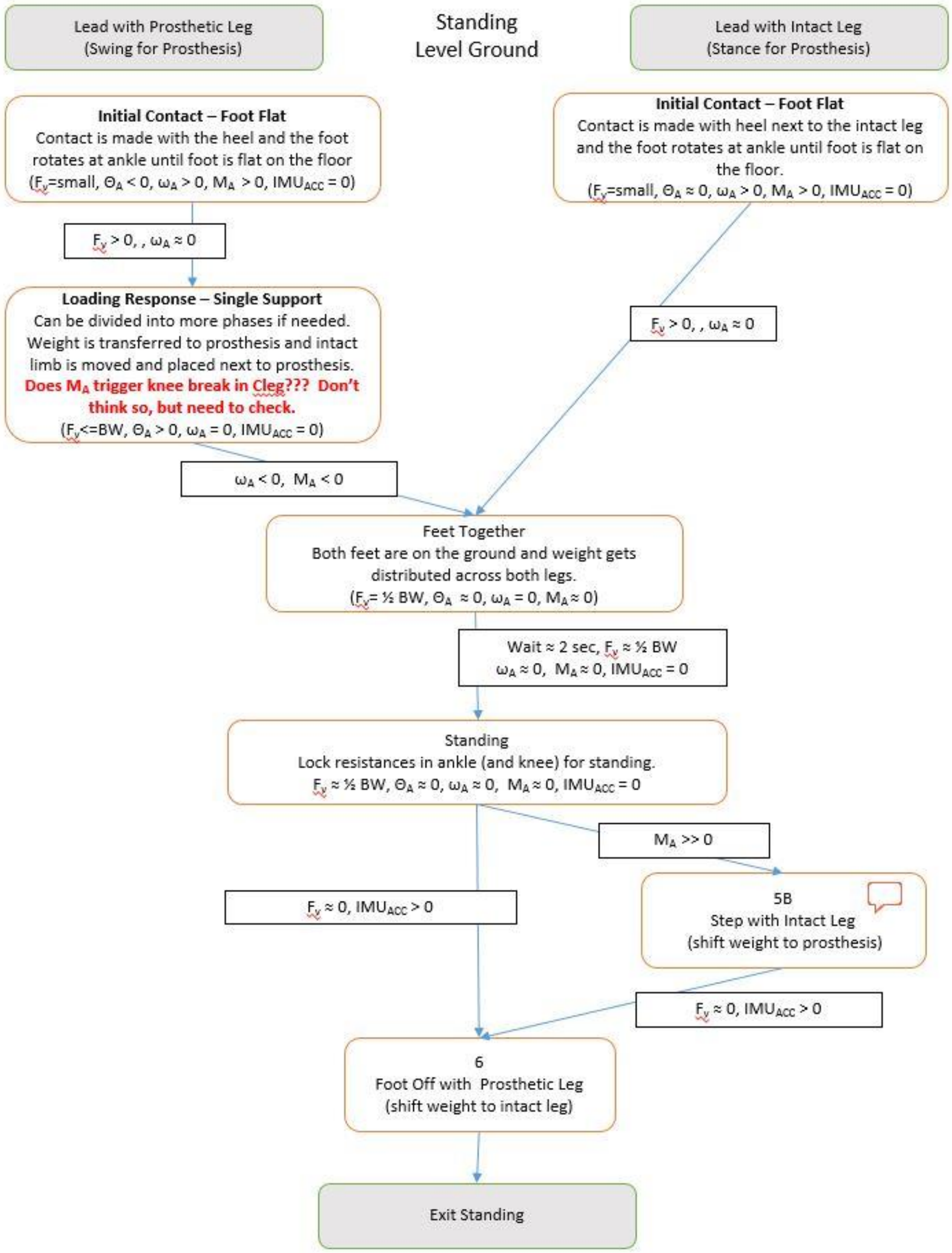
- In order to achieve the aforementioned coordinated control, the stairs must first be detected and recognized by the system (see our coordination with Purdue Univ. on the stair detection system below). Our pursuit of multi-source data streaming (video, IMU, etc) to the arbitrator will effectively provide this recognition.
- The ankle moment should not trigger the "knee break" state in the C-Leg during descent (steps e-f above).



From: "Forward stair descent with hybrid neuroprosthesis after paralysis…" Bulea, Kobetic, Audu, et al

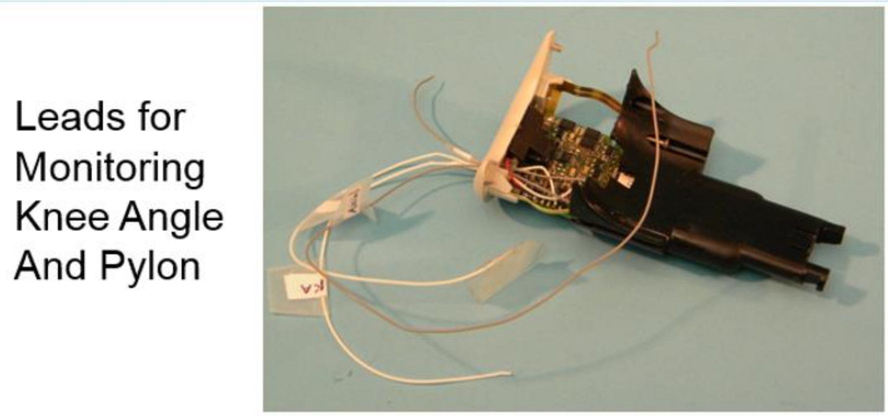
Other scenario based coordinated control requirements were identified, including jumping down, running, slope walking, level walking, stair ascent, squatting, standing, uneven terrain, weighted tasks, and ladder climbing, and the full details are included in Appendix A.

The updated interoperation control strategy is shown below. The initial interoperable control strategy will be developed using IMU sensors mounted to the foot, ankle, and upper thigh, along with a pylon load cell, however the algorithms will be iterated as the project matures to incorporate new sensors and improved functionality, allowing progressive but continuous development. The chart below shows the sensor input requirements and expected behavior of the leg during the aforementioned scenario. The IMU data will be used to derive the angle and rate of change information of the ankle and knee joints.



Also under this task, KCF has reviewed prior work conducted by LTI on the Otto Bock C-Leg. Knee angle and angle rate of change are two parameters, among others, which have been identified as necessary for the coordinated control algorithms under development. With the current LEGS sensor system, these parameters can be derived based on IMU sensors located above the knee and below the knee, however a knee angle sensor may already exist within the C-Leg. LTI has identified a magnet sensor, most likely a hall-effect type, located in the knee joint. It is unclear at this time whether this sensor indicates a single position of the knee joint or if it

reports the relative angle of the joint through rotation, however including this sensor may be a viable option for high rate knee angle measurement if needed.



C-Leg control board shown with knee angle leads

Late in Year 2, LTI began investigating the potential of measuring ankle torque on the College Park Odyssey hydraulic ankles indirectly by using a small electronic oil pressure sensor. When a load is applied to the Odyssey ankle, the oil within the hydraulic chamber undergoes a pressure transient that can be measured and then mathematically related back to the magnitude and direction of the applied load. Additionally, LTI is in the process of investigating the potential use of the Ipecs or SmartPyramid devices which can be quickly integrated into the prosthetic device and used to measure forces and moments at the ankle to allow for improved algorithm development.

C7: Update state machine for external sensors and clinical communications

- <u>All subtasks for Year 2 completed</u> – multi-user, multi-prosthetic oversite to be added in Year 3 as part of continued UI development
- <u>Version 1 UI development</u> was completed in Year 2. The LEGS UI includes:
    - Live sensor data streaming – wireless sensor data collected by the Arbitrator is streamed in real time to the cloud UI – the UI can display raw sensor data or calculated data representing derived information (such as knee angle, etc)
    - 3D patient avatar, showing gait and motion simulation using digital human models. The patient avatar can be rotated in 3 axes, and shows live motion data or replayed data recorded previously from the IMU sensors
    - LEGS UI dashboard display, sensor status indicators, activity statistics, and alerts & notifications, which allow the user to set Warning and Error levels for any sensor input or mathematical combination of inputs, such as low battery; excessive force; exceeding acceleration (falls, etc); exceeding joint angles, etc.
    - Built out database support and service provider integration for sending text based notification to users
    - Statistical software indicators providing a percentage of alarm or state time over a fixed time period, allowing the data reviewer or user to be alerted to the duration of an alarm or alert state
    - Software support (custom data connectors) was added to the UI back-end for generic sensor types
- <u>KCF developed enhanced wireless data compression algorithms</u>, and large batches of data were evaluated

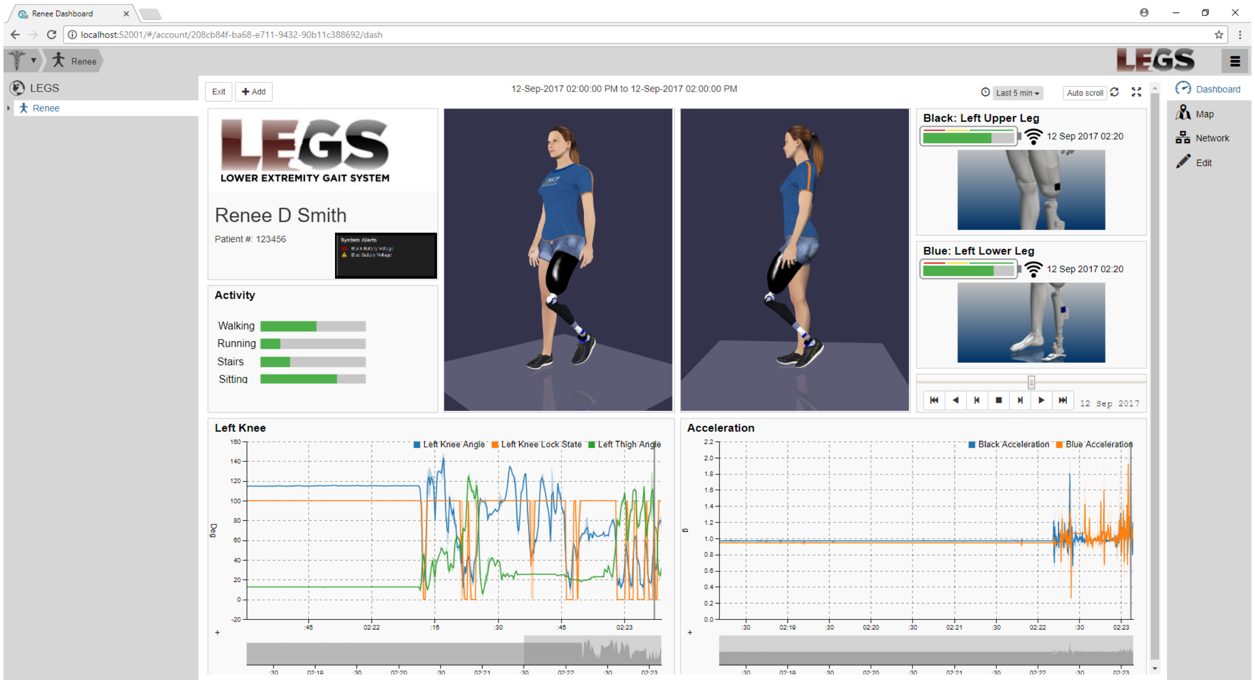
- Development was completed on the communication and data handling processes for multiple IMU sensors, allowing live sensor data to stream to the data collection cloud service. Data collection, processing, transmission, and calibration capabilities were added and built into the wireless sensor and Arbitrator boards.
- PDCP (the open, universal Prosthetic Device Communication Protocol) was discussed at length with Todd Farrell of LTI and Blair Lock of CoApt – use of PDCP has declined since this topic was proposed, so other options were developed for Year 3 implementation
- An initial concept for a universal prosthetic system translator was developed based on feedback from LTI and CoApt, which should address the barriers to PDCP. This concept will be matured and proposed for Year 3 development under this project
- Arbitrator software programming was matured, formalizing software installation to the arbitrator. A VPN (Virtual Private Network) was also integrated allowing the Arbitrator to be updated and serviced anywhere, regardless of its location

Task C7 Detailed Description

All C7 tasks for Year 2 have been completed with the exception of "Add multi-user, multi-prosthetic oversight to LEGS UI". Multi-user oversite will be added in Year 3 as the UI is matured.

Sensor data streaming to the cloud collection service, numerous LEGS V1 UI features, and open protocol concept development were the main accomplishments of Task C7 in Year 2. The LEGS system is currently capable of retrieving all wireless sensor data streams, transmitting them to the cloud data-store, presenting the data in an analytical format, displaying a real-time or recorded data 3D patient avatar representing patient movements, and displaying and sending alerts based on preset thresholds.

LEGS UI:



LEGS User Interface, showing 3D patient motion avatar, sensor placement and status widget, activity statics, patient alerts and alarms, and configurable data displays showing knee angle, etc, and raw sensor data

The layout includes an informative dashboard with relevant “whole body” data indicators and alerts, rather than focusing on raw sensor data. The Dashboard contains the functionality to review and edit the location of the various LEGS sensors in the prosthetic system, general patient information, battery statuses, alerts, and statistical activity data. The Trend View contains the 3D animated patient avatar as shown above, along with relevant clinical presentation of body and prosthetic movement over time.

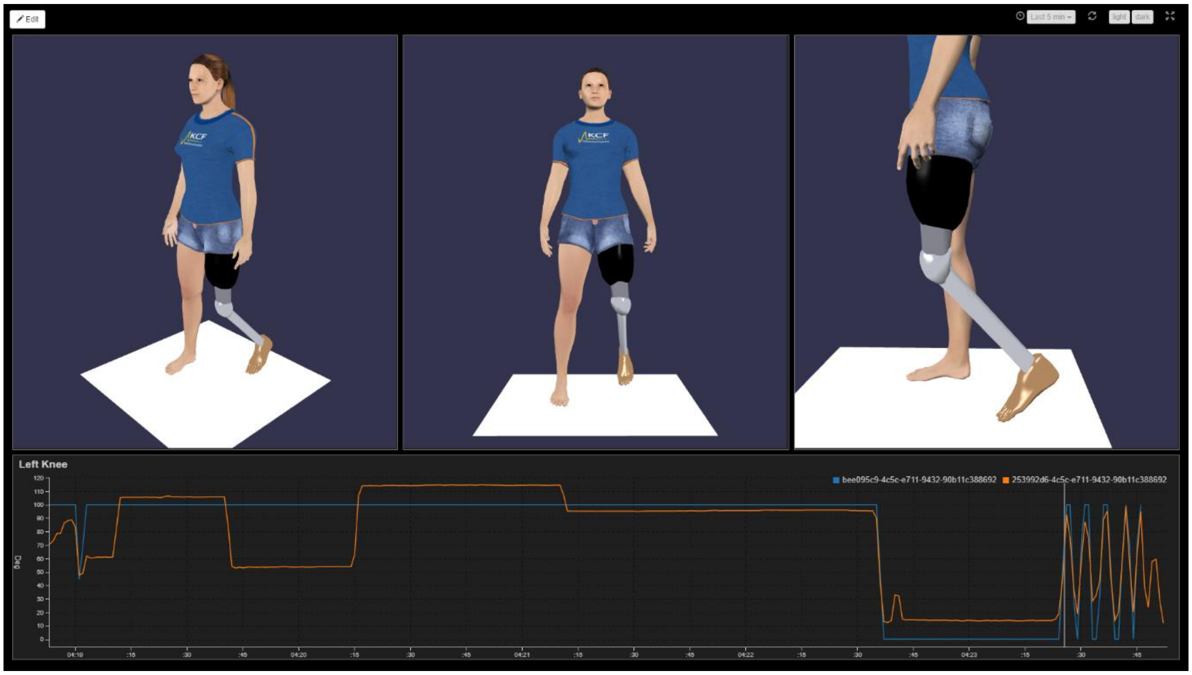
Controls were added to the time-series graph to allow more efficient selection of the time span of interest. The user can start with a large time span and view the average data points. This quickly shows periods of activity, and periods when the system is off-line. The user can then zoom in to see more detail and precisely analyze specific motions. By refreshing the data on a smaller timescale, the individual data points are shown instead of the average.

3D Patient Motion Avatar

The 3D patient motion avatar was developed to assist in detecting deleterious gait motions in conjunction with traditional charts. Although watching an avatar is more subjective than taking measurements from a chart, it is a more natural form of the data for the clinician. Patient motion can be animated from the data collected by the IMU sensors where the IMU data is converted into join angles which are then used to position the avatar. Live data or recorded data can used for the avatar, which allows the motion to be played back at various speeds to aid in the analysis. The avatar can be rotated so that motion can be viewed from any angle.

A “Player” control was also added; after the initial time position is set with the time-series graph, the Player control can be used to play the motion at a controlled rate. It includes buttons for

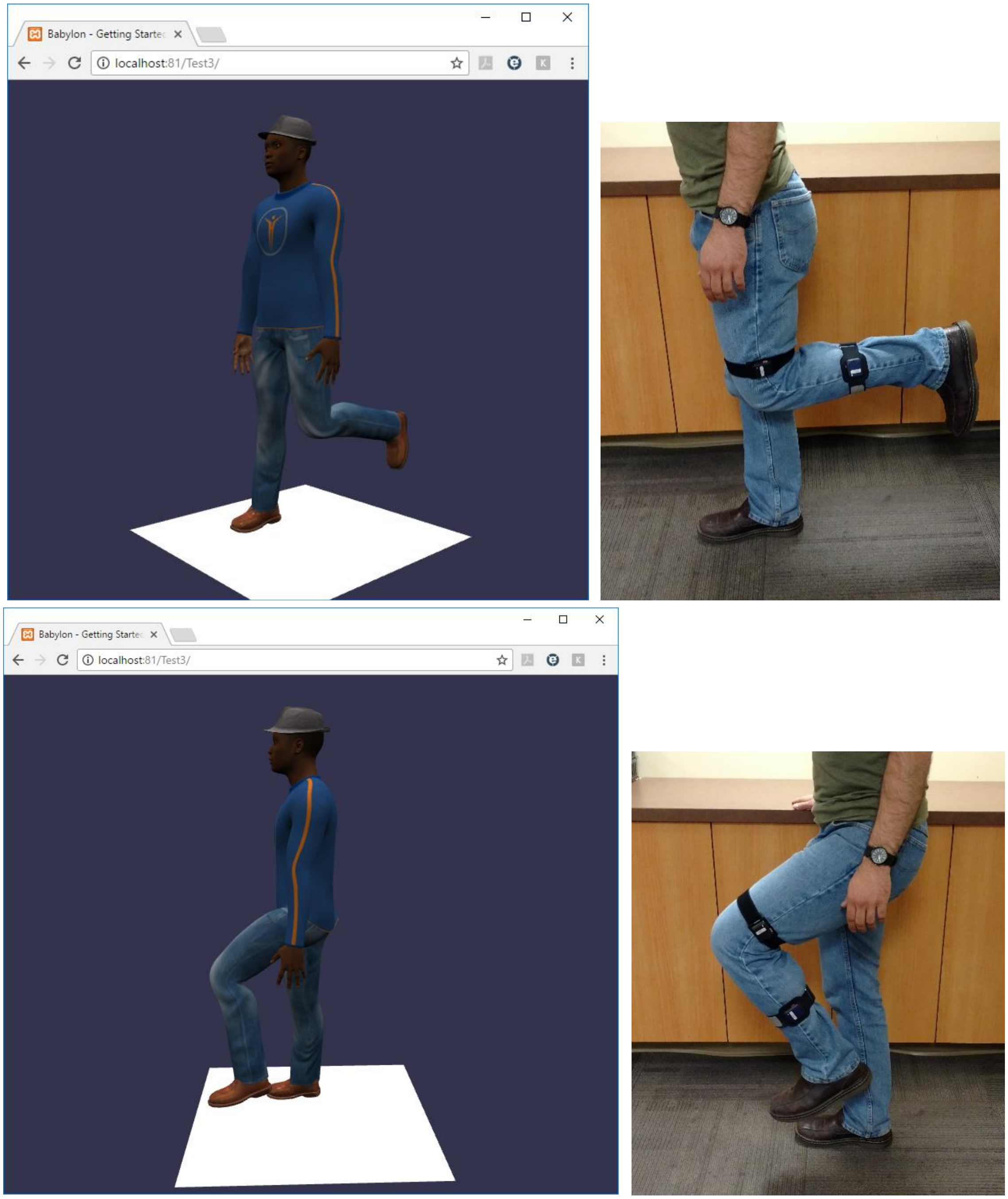
single-step, normal speed, and 2x speed as well as stopping. The motion can be played in both forward and reverse directions. A Shuttle slider is included which allows continuous, fine control of playback speed from x3 reverse to x3 forward. This allows the motion to be closely examined, and allows efficient movement between sections of interest. The actual time that is currently displayed is also shown in the Player control.



Patient motion avatar demonstration based on recorded and live IMU sensor input - The display shows the avatar seen in multiple rotatable views with knee angle and knee damping states shown on the same graph.

The 3D patient model is created using the open-source human CGI program MakeHuman. This program allows the model to be customized and can accurately model the actual prosthetic wearer. For this demonstration (screenshots are shown below), live data was used. This allowed simple verification that the avatar was following the movements of the subject.

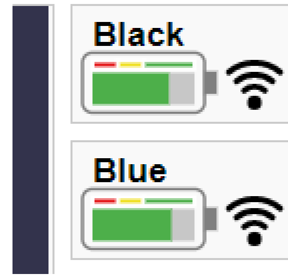
For the demonstration, one IMU was attached to the thigh, and a second to the ankle. The knee angle was calculated based on the relative angle of the two sensors. The angle at the hip was calculated assuming the torso is upright, and therefore only relied on the thigh sensor. An additional IMU on the torso can correct this assumption in the future.



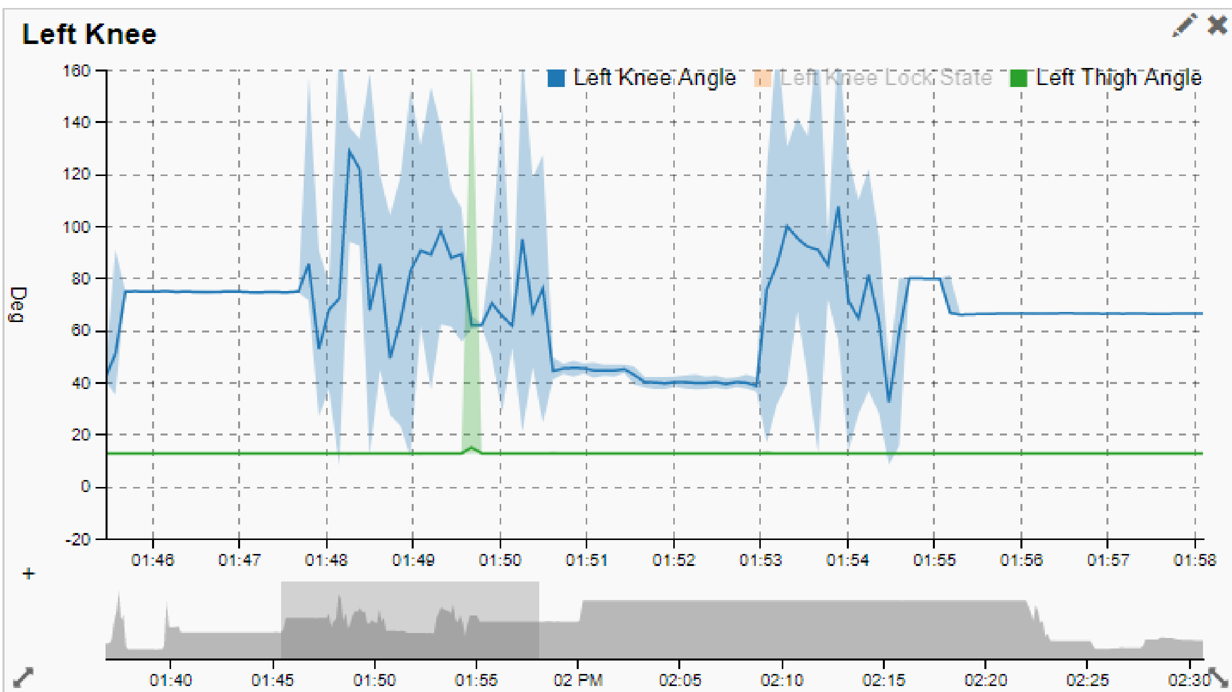
Static screen captures from the dynamic live avatar/user testing. Using an open source CGI animation program (MakeHuman) IMU data controls the movement of the CGI avatar in real time. This will allow remote viewing of the patient's kinematic movement either in real time or from recorded data

A "Node Status" widget was added to the user interface. This helps the user to ensure that all the nodes are functioning correctly prior to use. An on-line indicator shows that communication with

the node has been correctly established. The date and time of the last time that the node reported in is shown, which helps to diagnose lost communication problems. A battery level display is included, with warnings when the battery drops to a low level. The battery level is shown even when the node is not on-line. A graphic is also provided to show the correct placement of the node. The nodes are color coded to provide for simple identification. The widget is dynamically configured based on its size, so it can be used to just display the battery and connection status on an overview screen, or to display all of the information on a setup screen.

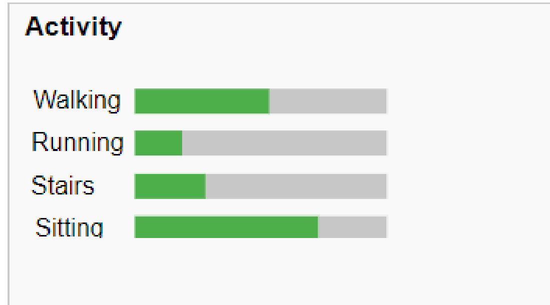


The time-series graphs were added to improve working with large data sets. These graphs can show raw sensor data or derived motion data (knee angle, etc). The time scale is adjustable, allowing the viewer to view large spans of time or zoom in to very short spans, and the graph automatically adjust the display resolution.

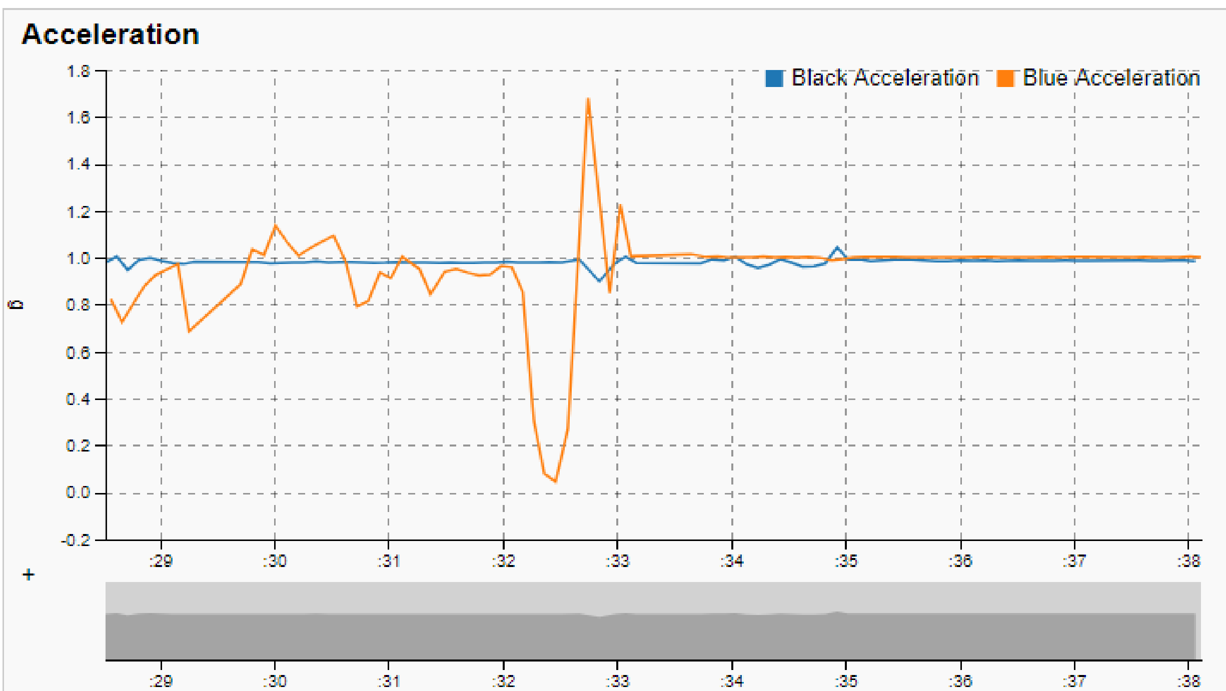


Time-series graphs showing knee motion, for example. The data display handles large data sets and automatically adjusts the data resolution when zooming in and out

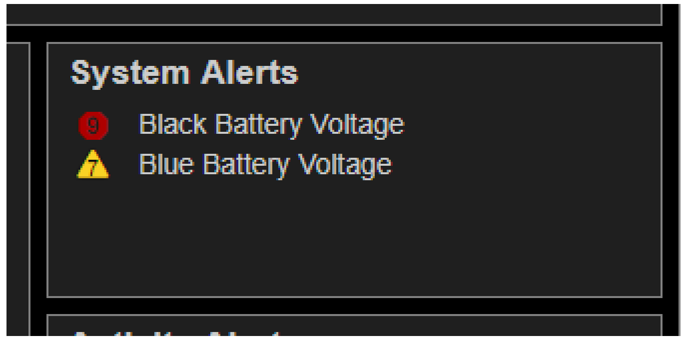
A stand-in widget for an activity monitor was created. This is a static display that is a visual representation of a planned activity monitor. The activity monitor would measure durations of typical activities within a time period. The back-end work of identifying the activities based on sensor input has not yet been developed.



The wearable Inertial Measurement Unit nodes were modified to allow the capture of acceleration magnitude prior to normalization. The normalized acceleration vector is used to calculate the angle of orientation, and thus the joint angles. Acceleration prior to normalization can be used to detect falls or other conditions that contribute to injury. Falls are characterized by a period of very low acceleration followed by a period of high acceleration. In the graph below, the orange line illustrates a fall from about 2 feet in height.



A simulated "fall" from IMU data- Very low acceleration followed by a high acceleration spike from dropping a sensor. This information will be used to setup user alerts to indicate falls and potential injury



Alerts feature were also developed in the UI. A new widget was developed to display Alerts that occurred in any chosen time span. This is relevant to things alerting the user to a low battery level, excessive force on the prosthetic, exceeding acceleration limits due to a fall, or exceeding recommended joint angles. The new widget shows a summary of the most recent alerts, quickly showing the user if there is a need to dig deeper.

The Alerts widget can be configured to only show alerts from certain indicators. This allows one widget to be used for system status alerts such as battery level or signal strength, and a second to be used for activity alerts such as fall detection or excessive force. In the future, there can be separate views for the clinician and patient, with only the relevant views displayed for each.

Arbitrator Data Streaming:

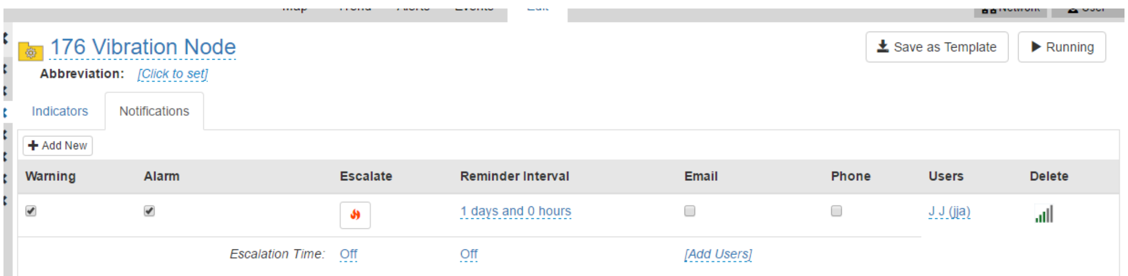
Initially in Year 2, extensive work developed the software needed for wireless data streaming into and out of the Arbitrator, along with the support required for reading sensor data into the clinical UI software. Support was built out in the arbitrator for the IMU's custom wireless protocol and collection server code was developed to run on the Edison platform, then ported to the Pi Zero W Arbitrator. This collection server code consists of a custom build of Linux and Mono Project (open source .NET framework). Additionally, support was added to the user interface software allowing generic sensor types, which are used to display the various outputs of the IMU sensors. In-memory caching framework for data ingestion was also developed, which makes high speed data acquisition from the IMU possible within the cloud framework and allows larger scale out of current cloud architecture.

Wireless Data Compression Development:

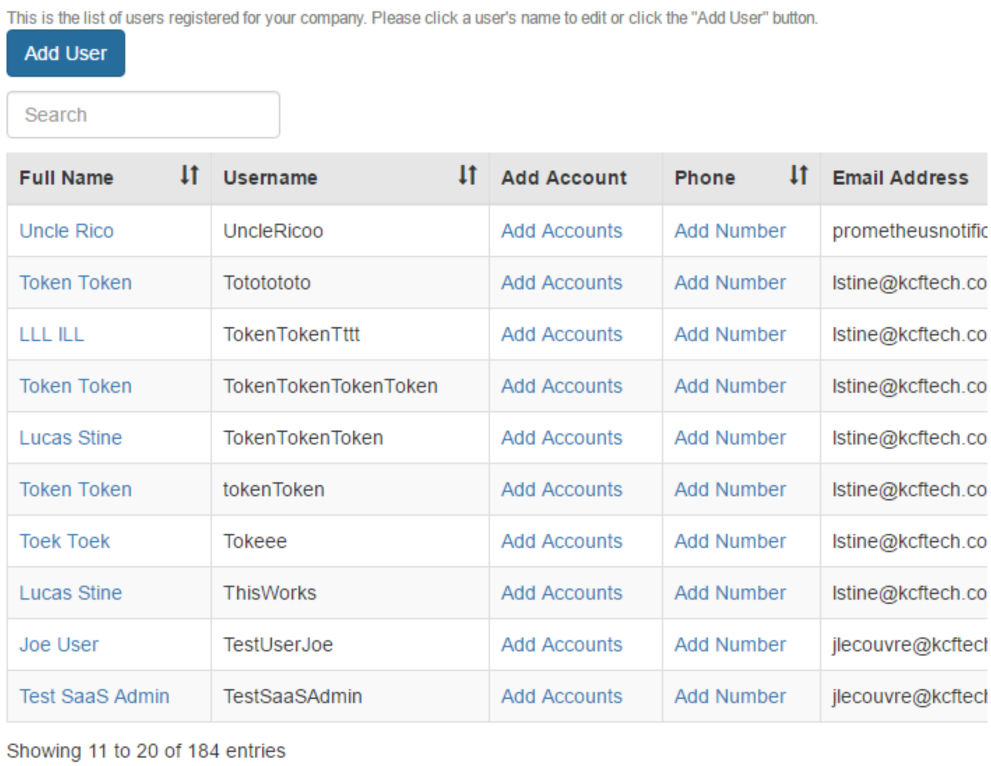
The data compression scheme required to efficiently and quickly stream high volumes of IMU, video, and general prosthetic sensor data in and out of the wireless Arbitrator was developed this year. The adaptive frequency compression scheme was upgraded to enforce specified error thresholds in the frequency domain to further reduce compression induced errors. In the remaining stages of the data compression algorithm development, a large number of data sets were tested and validated.

Text Notifications:

The LEGS database and service provider integration was built out to allow sending text based notifications. In application, this feature of the software interface allows critical information, such as battery life, high shock loads, and deleterious prosthetic use or health conditions to be communicated to clinicians and/or users as they occur for immediate action or correction. In addition, on time/off time/alarm time indicators were created in the software interface, allowing clinicians to set alerts for active and inactive timespans of the prosthetic system or sensors.



Notification interface menu, allowing the user to configure the alarm and notification settings



This is the list of users registered for your company. Please click a user's name to edit or click the "Add User" button.

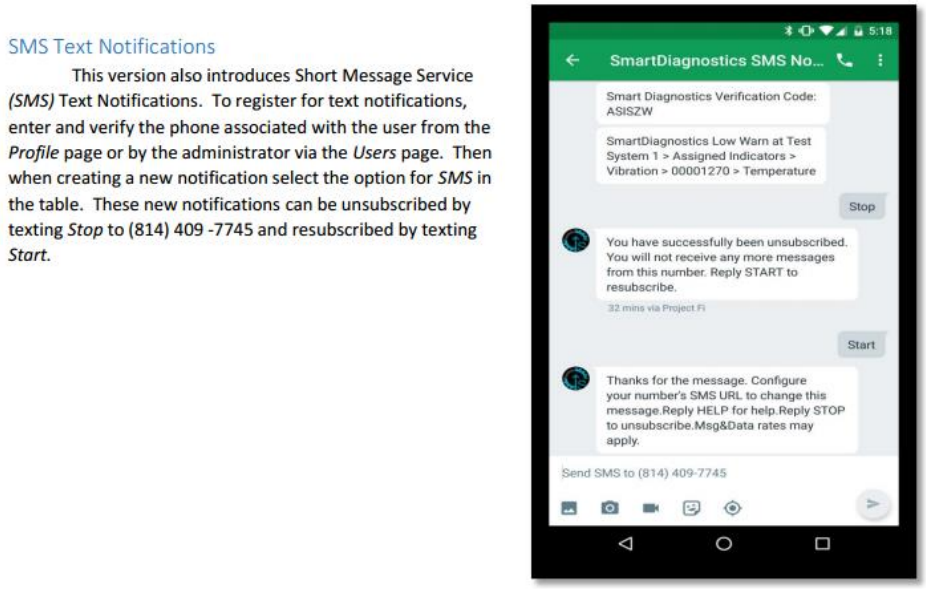
Add User

Search

Full Name	Username	Add Account	Phone	Email Address
Uncle Rico	UncleRicoo	Add Accounts	Add Number	prometheusnotific
Token Token	Totototototo	Add Accounts	Add Number	lstine@kcftech.co
LLL ILL	TokenTokenTttt	Add Accounts	Add Number	lstine@kcftech.co
Token Token	TokenTokenTokenToken	Add Accounts	Add Number	lstine@kcftech.co
Lucas Stine	TokenTokenToken	Add Accounts	Add Number	lstine@kcftech.co
Token Token	tokenToken	Add Accounts	Add Number	lstine@kcftech.co
Toek Toek	Tokeee	Add Accounts	Add Number	lstine@kcftech.co
Lucas Stine	ThisWorks	Add Accounts	Add Number	lstine@kcftech.co
Joe User	TestUserJoe	Add Accounts	Add Number	jlecouvre@kcftecl
Test SaaS Admin	TestSaaSAdmin	Add Accounts	Add Number	jlecouvre@kcftecl

Showing 11 to 20 of 184 entries

User list for text based notifications, which may include the user, clinician, or anyone requiring notification



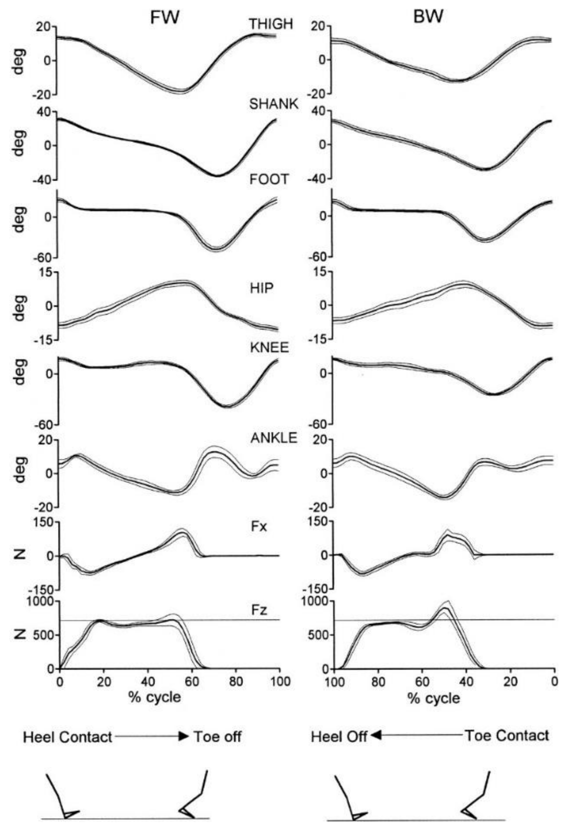
SMS Text Notifications

This version also introduces Short Message Service *(SMS)* Text Notifications. To register for text notifications, enter and verify the phone associated with the user from the *Profile* page or by the administrator via the *Users* page. Then when creating a new notification select the option for *SMS* in the table. These new notifications can be unsubscribed by texting *Stop* to (814) 409 -7745 and resubscribed by texting *Start*.

Example of text based notification

Gait Analysis:

Research was also conducted into clinical use of kinematic data to correct or aid in improving gait and setup lower limb prosthetics, specifically the best way of displaying data. Traditionally, visual observations are used to adjust prosthetics to the user's gait motion, however data collection systems are increasingly being utilized for this purpose, albeit typically in a research environment. Existing prosthetics, such as the Otto Bock Genium, utilize onboard IMU, load, and moment sensors to adjust gait cycle dynamically, and research with retrofit sensor suites has been conducted, however few if any prosthetic systems exist which combine these functions and contain integral sensors which stream gait cycle data for diagnostic use.



Analysis of able-bodied gait cycle measuring limb angles and loads - from: "Motor Patterns for Human Gait: Backward Versus Forward Locomotion by R. Grasso, L. Bianchi, F. Lacquaniti

Results have shown "significant improvement in lower extremity joint kinetics symmetry when using [a] microprocessor-controlled knee" ("*Gait asymmetry of transfemoral amputees using mechanical and micro-processor controlled prosthetic knees", K. Kaufman et al)*, where these studies utilized 3D gait measurements to calculate joint symmetry throughout the gait cycle.

PDCP / Open Communications Protocol:

During Year 2, KCF personnel had multiple meetings with Todd Farrell from LTI and Blair Lock from CoApt with regard to PDCP, the proposed universal and open Prosthetic Device Communication Protocol. PDCP was developed at the University of New Brunswick, envisioned as a method of achieving CAN-based universal communications between multiple manufacturer's prosthetic hardware, and was written into the proposal for this project circa 2015. Since the development of PDCP however, few if any prosthetic device manufacturers have investigated using this protocol, with only LTI and a handful of research organizations implementing it for ease of testing.

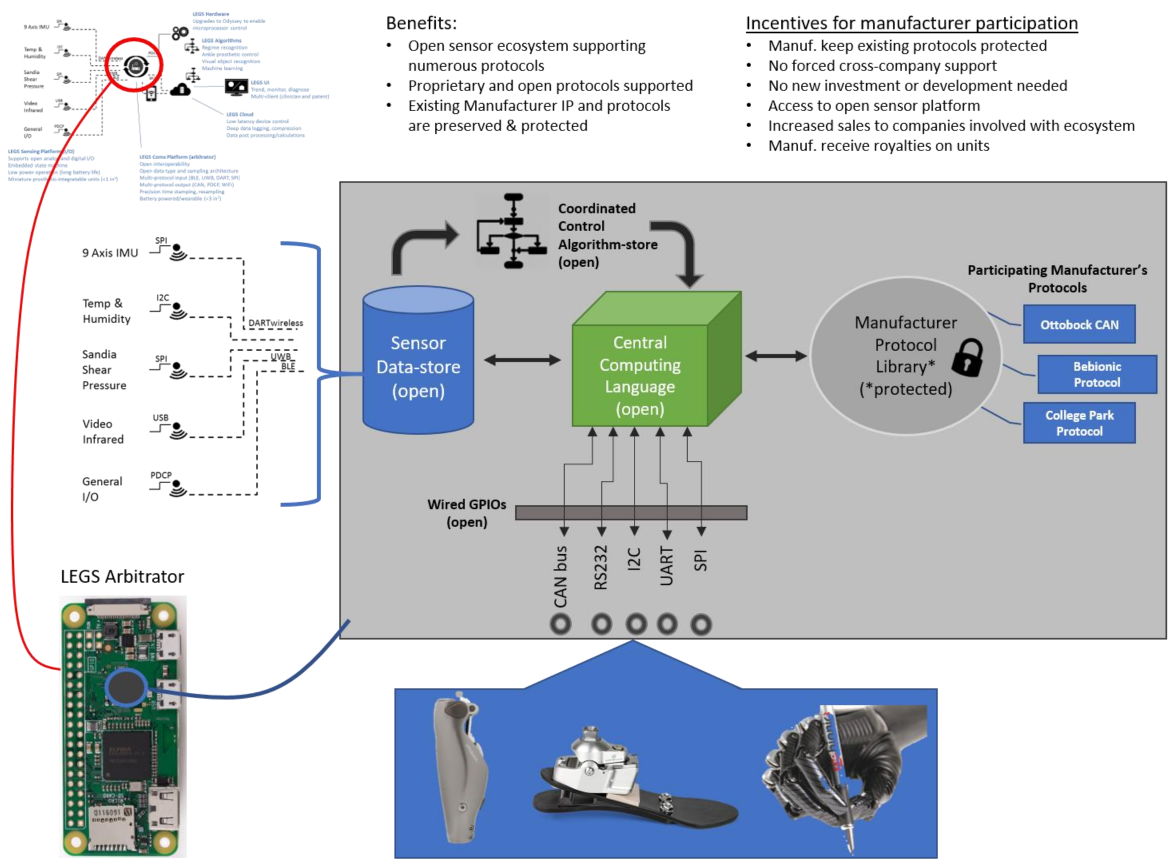
Mr. Farrell explored the use of PDCP in current prosthetics devices at the MyoElectric Control Conference, and KCF discussed the current state of PDCP with Mr. Farrell and Mr. Lock in order to ascertain the status, adoption, and implementation of PDCP for this project. Currently, PDCP has fallen out of favor with device manufacturers and is not envisioned to be adopted at any level. Based on this information, it is apparent that the use of PDCP has declined and other options should be considered for this project. Personnel from KCF, LTI, and CoApt then discussed and

developed a new strategy for an open communications protocol which addresses most of the barriers to PDCP.

Mr. Farrell and Mr. Lock provided their opinion to explain non-adoption of PDCP, with reasons including:

- <u>PDCP proved too complicated</u> – Knowledge of CAN bus protocols within existing manufacturers is lacking, presenting a significant barrier to development
- <u>Large investment required</u> – Existing manufacturers would be required to invest significant resources in PDCP implementation with no foreseeable ROI
- <u>No incentive for manufacturers</u> – No discernable un-tapped market exists; no lost-opportunity risk; no major industry players leading the way
- <u>Prosthetics manufacturers' trust</u> – No trust between manufacturers to disclose IP or develop interoperability; no neutral party to broker interoperability
- <u>Liability and support risks</u> – Manufacturers don't want to be held accountable for user complaints or damages resulting from interoperability with other manufacturer's hardware

Taking these barriers into consideration, a new concept for interoperable communications was developed. KCF proposes to further refine this concept before starting development in Year 3 of the project. The proposed translator would consist of four major pieces: the sensor data-store; central computing language; coordinated control algorithm-store; and the manufacturer protocol library.



Proposed concept for an interoperable prosthetics communications system to replace PDCP

The first component, the sensor data-store, would serve as an open database for all incoming sensor data. Any device or algorithm on the system would have access to the data-store for use.

The central computing language would serve as the over-arching arbiter of the translator, based in an open language like Python, and handle the GPIOs, coordinated control algorithms, communications with the various protected protocols, etc.

The third component is the coordinated control algorithm-store, an open code repository in the central computing language which handles interoperation of prosthetic hardware based on kinematic motion.

The last component would be a manufacturer protocol library; this being the only closed or protected portion of the translator, it would consist of embedded firmware capable of translating various participating manufacturers protocol I/Os to the central computing language. Keeping this code closed would alleviate concerns among manufacturers over IP release and unauthorized use of their systems.

The benefits of the translator system include:

- Open sensor ecosystem supporting numerous protocols
- Proprietary and open protocols supported
- Existing manufacturer IP and protocols preserved and protected
- Interoperability between participating manufacturer’s prosthetic system achieved

Incentives for manufacturer participation include:

- Manufacturers keep existing protocols protected
- No forced cross-company support
- No new investment or development needed from manufacturers
- Access to the open sensor platform and sensor data-store
- Access to the coordinated control algorithm-store
- Increased sales for companies participating in the ecosystem
- Potential royalties paid to participating manufacturers

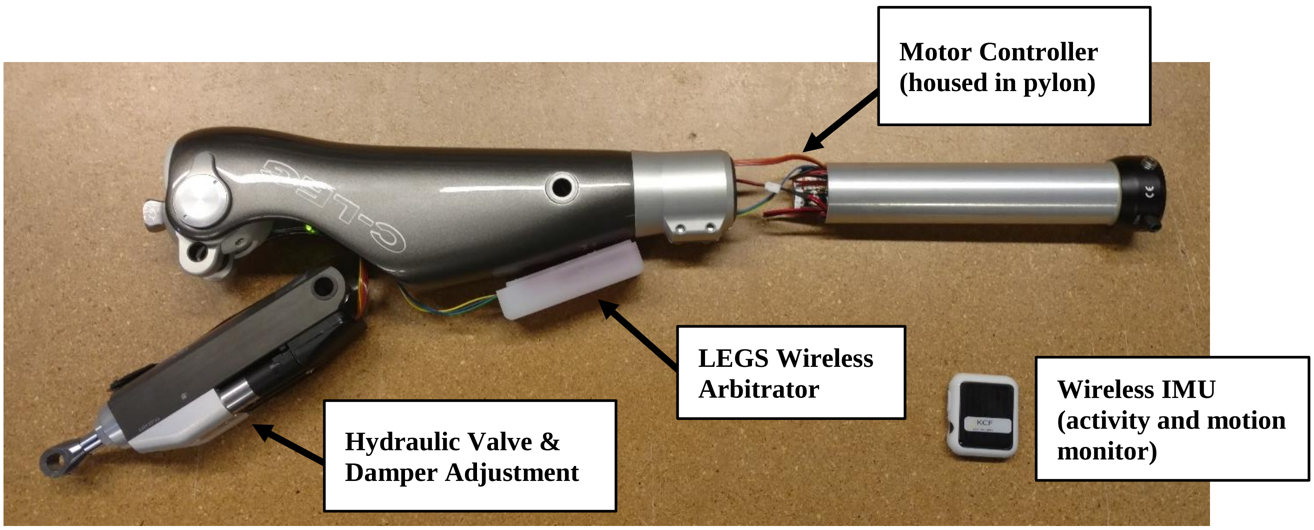
C8: Prototype and test components and integrate external sensors platform

- All subtasks for Year 2 completed
- Multiple IMU sensor and the pylon load sensor inputs tested with LEGS arbitrator and cloud data streaming
- The Bluetooth pylon load sensor was successfully tested and transmits accurate load information via Bluetooth UART to the Arbitrator and cloud data service
- Prototype knee/ankle system successfully tested for sensor based, biometrically controlled adjustment
- Odyssey foot IMU and electromechanical adjustment integrated with prosthetic foot hardware, foot dorsiflexion and plantarflexion control tested with PPM control signals
- IMU sensors tested on able leg for data streaming rate and CGI representation
- IMU battery life testing was conducted, resulting in IMU battery life of approximately 18 hours
- V2 Arbitrator battery life tested, resulting in 1.5 days of run time on a single 2000mAh battery
- The Socket Monitoring SBIR extension is coming to conclusion, which will then transition to the LEGS platform. KCF and Willow Wood have produced a molded

prototype with integrated sensors for testing, which will then be delivered to University of Pittsburgh's School of Health and Rehabilitation Services
- Additional IMUs were manufactured, allowing full coverage including sensors on the prosthetic foot, prosthetic lower leg, prosthetic upper leg, waist, and able leg

Task C8 Detailed Description

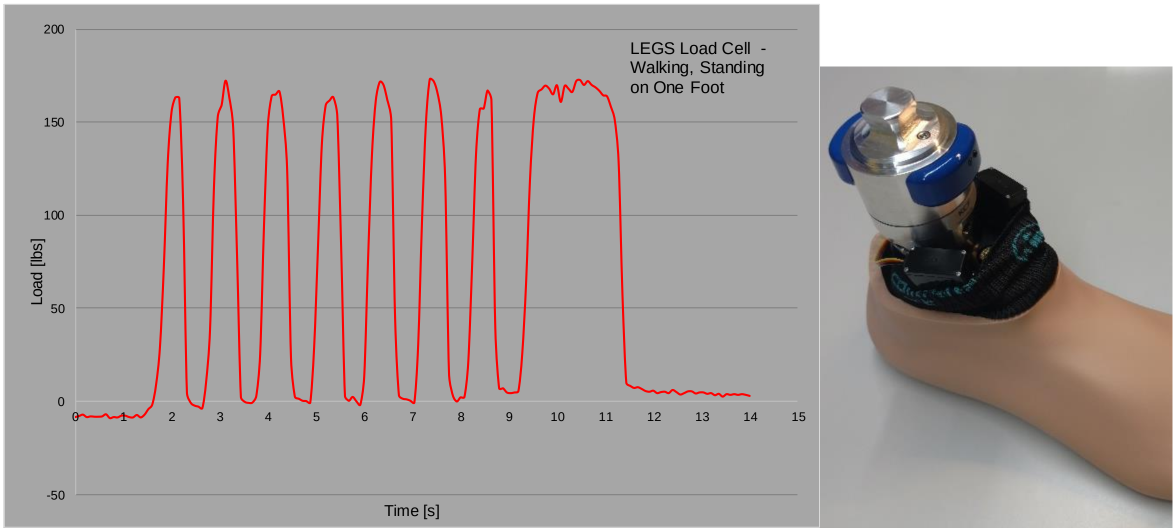
All C8 tasks for Year 2 are complete. The prototype prosthetic hardware components have been completed, sensor support for the forthcoming socket sensors has been added to the Arbitrator and UI, Odyssey ankle control has been demonstrated, and the high bandwidth optical sensor functions as a stand alone device.



Components of the LEGS technology demonstrator on a C-LEG knee

The C-LEG and Odyssey K3 foot were chosen as the platform for the prosthetic control technology demonstrator in order to illustrate the advances afforded by this research program over-and-above a state-of-the-art prosthetic device. The currently available C-Leg and Odyssey foot have different damping states which are available for selection manually via a handheld remote transmitter (in the case of the C-Leg), similar to a garage door opener. In order to show the capability of an adaptable "smart prosthetic" device standardized under this project, the adjustment for the knee and foot damping will be handled automatically by the LEGS arbitrator based on wireless input from the LEGS sensor suite. When certain activities or motion states are detected by the data processor in the arbitrator, the appropriate valve damper setting will be set, requiring no manual user input.

The image above shows the C-LEG modified with a simple H-bridge motor control circuit (stored in the lower pylon), which is controlled by the arbitrator based on the data streaming via DART wireless from one or multiple wearable or prosthetic mounted IMUs.

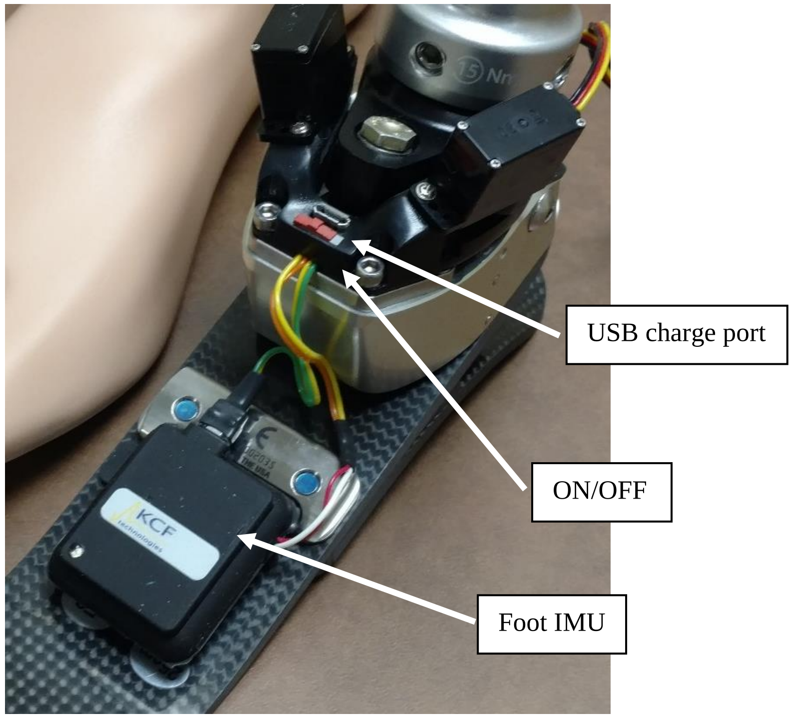


Data collection from the LEGS wireless BLE pylon load sensor

The Bluetooth pylon load cell was assembled and tested. The pylon load sensor and wireless BLE datastream was tested for data capture simulating multiple steps at full weight (~168lbs) and standing at full weight. The data was collected from the Python script running on the data Arbitrator and is shown above.

Under this task an additional IMU was fabricated for integration with the Odyssey foot. The IMU is mounted to the foot and is covered by the cosmetic foot cover. A remote ON/OFF switch was also wired into the servo mount along with a USB charge plug to provide remote on/off and charging while the foot cover is in place.

The functionality of the electromechanical foot adjustment was also tested with a hand-held servo controller – digital servos were selected to allow adjustment of end points, rotation speed, rotation angle, and center point. Currently the arbitrator outputs to control the servo motors have been implemented.



External sensors integrated with the Odyssey K3 foot

Under this task, the wristband IMUs were run from full battery to low voltage shutdown at a constant 10hz data transmission rate. IMUs were turned on at 5:06pm 7/10. IMU #20000014 ran until 11:28am 7/11, and #20000015 ran until 11:18am 7/11 giving an IMU battery life of about 18 hours.

Ideally, 24 hours or more of battery life could be targeted to allow a buffer before requiring recharge. If needed, battery life could be extended by any or all of the following: increasing the battery capacity and housing size (increasing battery thickness would be ideal); implementing a sleep state in the microprocessor to automatically halt and resume wireless data transmission in low activity states (turns off when sitting, left on overnight, etc); decreasing the power output of the wireless transmission (currently transmitting for long range use, however the IMUs will never be far from the arbitrator when worn on the body), or replacing the battery protection and charging circuit with a custom or COTS low energy charge/protection circuit.

Battery life of the 2000mAh arbitrator battery module was tested running the Pi arbitrator continuously, lasting about 1.5 days. It is expected that additional power will be required to drive the hardware control motors, however it is possible to reduce arbitrator power consumption by sleeping or suspending unneeded resources. Power efficiency can be increased by programming automated sensor sleep states during periods of low activity (such as sitting, driving, sleeping, etc). In these scenarios, the IMU sensors will detect that the user is idle and put the arbitrator and sensor radio systems in a low power sleep state, preserving battery life by only polling for data once a second or less.

Socket Liner:

The extended work period for SBIR DHP13-017 “Advanced Sensor Integration for Prosthetic Socket Monitoring” concluded in Sept. 2017, allowing the transition of technology to the LEGS platform.

Prototype socket liner units made under SBIR DHP13-017, which will be tested and integrated with the LEGS platform in Year 3

KCF Technologies has produced a prototype unit fully integrated into a commercial socket liner system produced by Willow Wood of Mt. Sterling Ohio. Once final functional testing is completed, the unit will be delivered to The University of Pittsburgh's School of Health and Rehabilitation Sciences for evaluation of fit and functional testing. Results of this testing will provide input to the final design revision. These units will support commercialization and technology transition into market.

C. Sensing Thrust Aim

Goal: Define patient-centric high bandwidth sensors and platform for optimal human-device interoperability, and demonstrate key capabilities in hardware demonstrations (annual milestone).

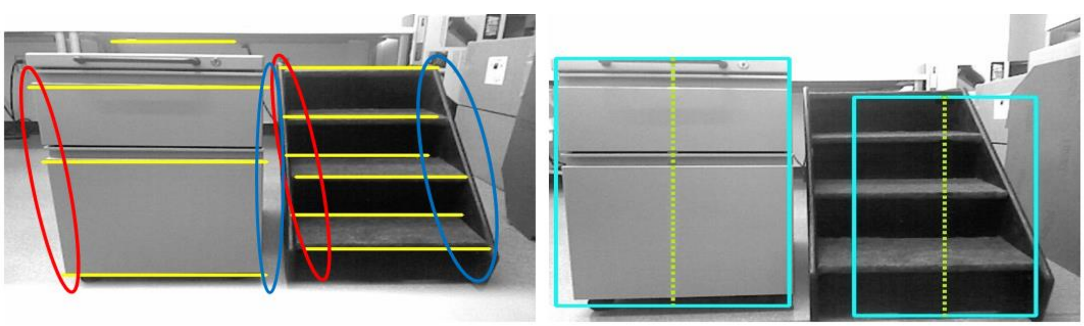
S5: Define high bandwidth sensor and platform design specification

- All subtasks for Year 2 completed
- Development of the high bandwidth optical sensor was completed in partnership with Purdue University - optical recognition of various descending and ascending real-world stairs was executed, with the algorithm successfully identifying different staircases
- Purdue University's complete report is attached in Appendix D
- Integration of the smaller Asus Xtion2 RGB and depth camera was completed – the Linux, OpenNI and OpenCV3 software was successfully updated to function with the Purdue Univ. stair detection algorithms
- Body mounting scheme was completed for the Asus camera, consisting of a belt mounted camera

Task S5 Detailed Description

In partnership with Purdue University, smart prosthetic video-obstacle detection work was completed for detecting ascending and descending staircases. The complete report from our Purdue University partners is included in Appendix D.

Left: Depth image showing the detected edges from the software algorithm, Right: these edges match the parameterized model. And hence the algorithm declares that stairs are detected



Left: Refined edge detection image from Stage 1. Red and blue ellipses indicate x coordinates of left and right end points; Right: Rectangle drawn around group of detected edges

The object detection algorithm first determines all relevant straight edges from the RGB image, then groups the edges into different sets of parallel lines. The edges whose ‘x’ coordinates of their end points are the same or closely located (shown by the red and blue ellipses) are assigned to the same group of parallel lines. Therefore the edges of the stairs are classified into one group of parallel lines while the edges of other objects are grouped into another set. After this classification is complete, a virtual rectangular boundary is drawn around these sets of edges. The depth values of all the points along the vertical central axis of the rectangles (shown by the green dotted line) are measured using the depth image. If the depth of these points gradually decreases (i.e., the distance from the camera becomes smaller) as we move from the top edge of the boundary to the bottom, then it can be assumed that the set of edges inside that rectangular boundary represent ascending stairs. A “stairs” icon is then drawn over the corresponding feature in the image and the distance to the lowermost edge is calculated and displayed.

Testing was conducted in real world environments. There are some added complexities to in using real stairs; they have different colors, designs (many of which are a set of parallel lines near the edges), textures and also added features like handrails. Many of the stairs in real world have more than two steps. So only the lower half of the depth image (as marked in Figure 3) was considered, where only the first few steps of the stairs are visible. Depth images were taken of 31 different stairs and depth values were extracted from the lower portion from each of the images. These depth values show a large gap or spike at the location of the edges of the steps. These spikes are taken as the key features for the analysis along with their respective depths and the distance between them. A parameterized model of the stairs is created using these features.

While working in real time, the algorithm looks for these spikes in the lower portion of the depth image. Whenever it finds two consecutive spikes, it calculates their depths and also the distance between their locations. This corresponds to the height and width of the steps (if the camera is really looking at a stair). The program then calculates the depth of the region between the two spikes (which should be almost uniform in case of a stair). These values are then compared with the parameterized model. If they are within some predefined thresholds, then the program decides that it is really looking at a stair and reports that the stairs are detected and the distance from the stairs.

Software: OpenCV 3.0. (Open source C++ based library for image processing, upgraded version), OpenNI (framework for Depth cameras), SensorKinect (open source package)

Downsized Camera:

Work during Year 2 was conducted to identify a smaller camera system to develop as the LEGS high-bandwidth object detection sensor and write the required C++ code for the OpenNI and OpenCV3 packages to interface with the compact camera. Although this work is critical in developing a compact visual object detection system (in this case for stairs), it also paves the way to introduce other optical sensor types into the LEGS environment as needed. As new and smaller optical sensor types are developed, OpenNI driver packages will be required to include these sensors in the LEGS operating ecosystem. KCF will continue to identify smaller optical sensor options as the LEGS system matures, and use the work conducted to date to incorporate the new sensors with the OpenNI software package.

The open-source stair detection algorithms developed by Purdue University relied upon an Xbox Kinect sensor for input. The Kinect sensor was disassembled to determine if a compact version could be made. The Kinect sensors consists of the depth, RGB, and IR sensors, audio output and amplifiers, and a motor for articulating the base of the camera. The circuit boards to control these devices fill the cavity of the sensor, making it unrealistic to downsize the Kinect sensor to a wearable scale. The algorithms behind the stair detection software were developed to be compatible with the Kinect sensor, fully utilizing the IR and RGB sensors as well as the OpenNI and OpenCV characteristics of the Kinect. These features play a major role in how the stair detection algorithm works with the Kinect. However, a smaller depth sensor camera which meets the software requirements to was found, the Asus Xtion 2 3D sensor.

Details

ASUS Xtion2 3D sensor

640x480p30 depth resolution
Supports high 5M(2592x1944) image resolution
4.3"x1.4"x1.4"
SDK Version Support: OPEN NI 2.2

Supports the following OS
Windows 32/64: Windows 8 and Windows 10
Linux 32/64: Ubuntu 14.04

Asus Xtion2 3D sensor – this sensor replaced the Kinect sensor for stair detection and functions as the LEGS ecosystem high-bandwidth sensor in a smaller package

Progress in linking the Xtion2 sensor and the stairs detection algorithm was successfully completed. The new Asus Xtion sensor is compatibile with the stair detection program from Purdue University, allowing full access of the sensors RGB and depth data. Success of this new sensor allows for a smaller and sleeker design for this prosthetics project. Though the cost of the sensor was slightly higher than the Kinect's, it shows greater resolution for both depth and RGB cameras and the smaller size allows the sensor to be body worn. Holding and mounting devices were also researched and purchased to allow the camera to be comfortably body-worn on a belt.

S6: Sensor platform detail design component integration

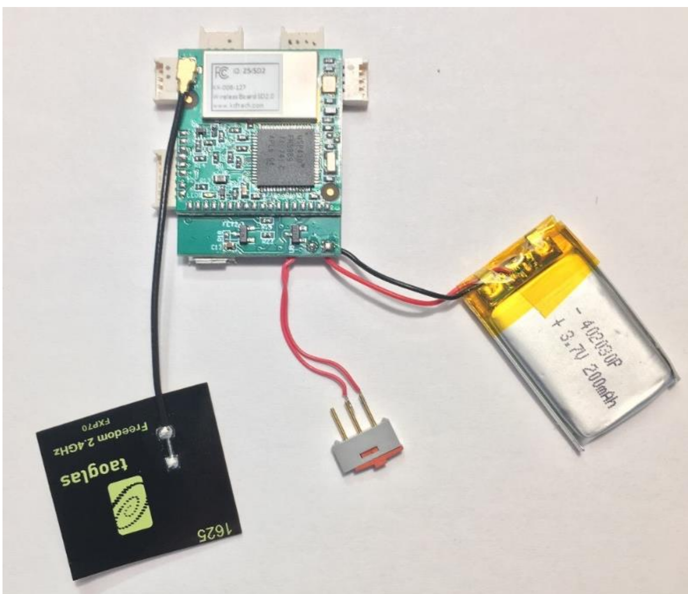
- All subtasks for Year 2 completed, with the exception of integrating socket monitoring (Socket Liner SBIR extension ended Sept 2017)

- Built electronics hardware for an open "agnostic" 4-20mA and 0-10V generic sensor input to the LEGS environment
- Multiple kinematic sensors (IMUs) integrated into LEGS environment
- Wearable sensor mounting and integration hardware purchased, designed, and fabricated for IMU kinematic monitoring sensors
- Socket sensors will be transitioned to the LEGS platform in Year 3 – the research extension of the Socket Liner SBIR has just concluded (late Sept 2017) and will be fully integrated into the LEGS platform in Year 3
- Open BLE wireless load sensor was completed and streams axial pylon load data via UART Bluetooth to the LEGS Data Arbitrator
- BLE load cell was redesigned for commercial effort, including a shorter stack height and direct integration with a 34mm C-Leg pylon. This will reduce the addition stack height of the load cell to under 1 inch

Task S6 Detailed Description

All S6 tasks for Year 2 have been completed, with the exception of integrating the socket liner sensors. This was due to an extension of the research period on the Socket Liner SBIR through the end of September, however the extension on that project allowed the maturation of the prototype units and socket sensors and electronics. Work on the socket sensors was conducted under its own SBIR, however it is reported here for information only as it pertains to the LEGS project. After completion of the research extension, the Socket Liner will enter market transition and will be integrated as a sensor component to the LEGS platform.

An updated socket liner was developed and sensor units sent to WillowWood for molding into four additional prototype units. The wireless board for the socket liner was redesigned and manufactured to include four connectorized analog inputs for the shear sensors and one I2C input for the temperature and humidity sensor. This sensor board can function as a universal wireless sensor input board for various other inputs to the LEGS ecosystem.



Work conducted under the Socket Liner SBIR as it pertains to LEGS – the new wireless board for the socket liner, including four analog inputs and an I2C input for shear and temp/humidity sensors, respectively. This technology will transition to marketing and be incorporated into the LEGS platform

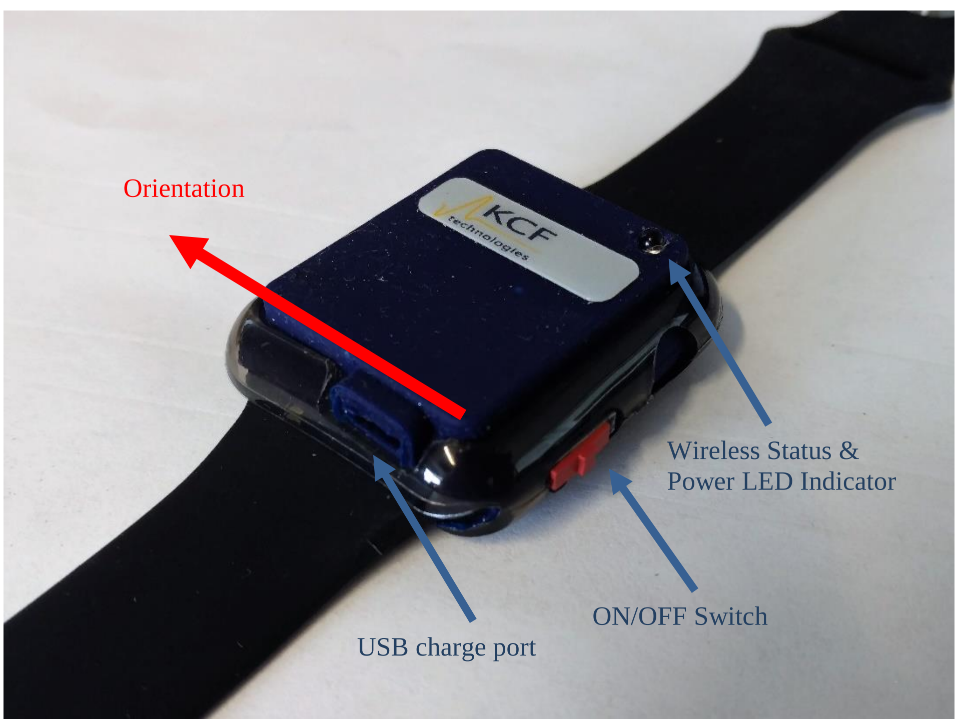
Development of the LEGS sensor package also included the addition of an open “agnostic” sensor to the LEGS ecosystem. This sensor platform is based on the DART wireless interface and allows any sensor providing 4-20mA or 0-10V analog output to be added to the LEGS wireless datastream. In keeping with the open source, multi-vendor approach allowing for auxiliary inputs to the prosthetic system, software support and hardware were completed for this open sensor. The agnostic sensor PCB design was completed and fabricated.

Wireless IMU Sensors:

Numerous wireless IMU (kinematic monitoring) sensors were developed for the LEGS project this year. The sensor consists of a DART wireless board and a STMicro LSM9DS1 chip with 9 axes of measurements (accelerometer, gyroscope, and magnetometer). The sensor package also includes a 150mAh LiPo battery, low voltage charge and protection circuits, On/Off switch, and flexible antenna. The sensor package is then overmolded with Smooth-On DragonSkin silicone in a custom mold, fully encapsulating the sensor from sweat and environmental hazards.

In total, five IMU sensors were built and provide full coverage of the prosthetic system and opposing leg. KCF has built out the data processing and connection software/firmware in the sensor/arbitrator communications hardware allowing multiple sensor inputs to be received and processed by the arbitrator. Each IMU node sends 3D orientation data (magnetometer), as well as corrected 3D acceleration and gyroscope data at a rate of 10Hz. The acceleration and gyroscope data is collected primarily to detect other events and will be utilized fully as the aforementioned coordinated control algorithms are implemented in Year 3. The orientation data from the IMUs are used to determine the position of the limbs in a global coordinate system (North, West, Up); since each node is presenting its orientation using the same coordinate system, the orientations can be directly compared.

The orientation vector chosen for our calculations is shown in the figure below; the positioning of the node is not sensitive to rotation about the limb, so the exact position can be chosen for comfort or convenience.



LEGS wearable IMU, kinematic monitoring sensor

Mounting and wearable integration hardware were also developed for the IMU sensors: the overmolding for the IMUs matches the shape and size of the 42mm Apple Watch, allowing the use of numerous COTS mounts and holders. In addition, silicone pigment was used to color code the IMU sensors for specific locations and uses. In the image below from left to right, a COTS armband and LEGS IMU sensor (which is large enough to fit an upper arm or thigh); wearable IMU wristwatch; 3D printed hardmount IMU case (can be fastened with M2 screws to any surface); and a bar or round mount holder for the LEGS IMU.

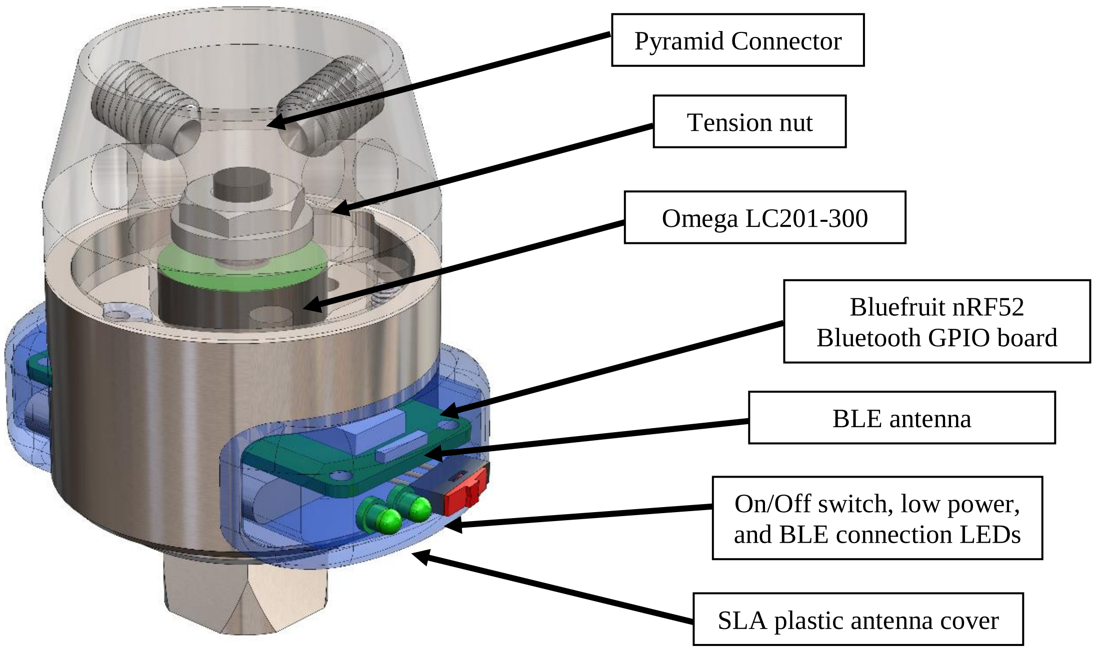
LEGS IMU mounting and attachment hardware, including arm/leg band, watch mount, hardmount case, and round/bar mount

Wireless Load Sensor:

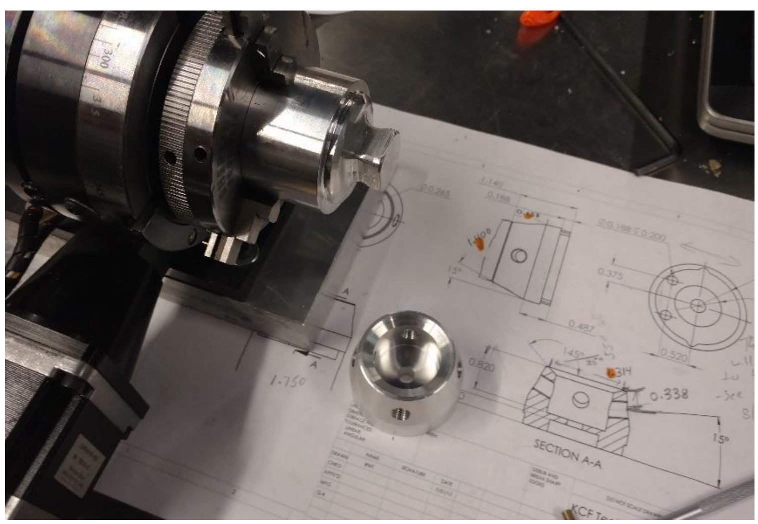
The LEGS open source Bluetooth pylon load sensor was completed this year. The load sensor uses the Adafruit nRF52 Bluefruit LE GPIO board, which is a simple, versatile, low power BLE GPIO board using the nRF52832 radio programmed directly via Arduino IDE. A Texas Instrument INA333 instrument amplifier was added to the board to boost low voltage load cell outputs. Programming includes BLE advertising, connection protocol, and polls the analogue-to-digital inputs for load cell voltage. The board then transmits the load cell voltage and its own battery voltage over BLE UART to the Raspberry Pi Arbitrator.

The Arbitrator program runs a Python script in the Raspberry Pi Linux environment using the BlueZ Bluetooth library to handle the BLE connection. Importantly, the arbitrator program can be set to connect to any available BLE device with UART service or a specific device with a provided MAC address (as in this case). This allows the addition of any BLE device with UART service to the LEGS sensor system.

Hardware for the LEGS open source load sensor includes aluminum pyramid connectors, SLA printed antenna and charge port covers, 500mAh LiPo battery, and an Omega LC201-300 load cell. The load cell is preloaded by the top nut to a calibrated tensile load, and all subsequent axial load is read as a decrease in tension on the load cell.



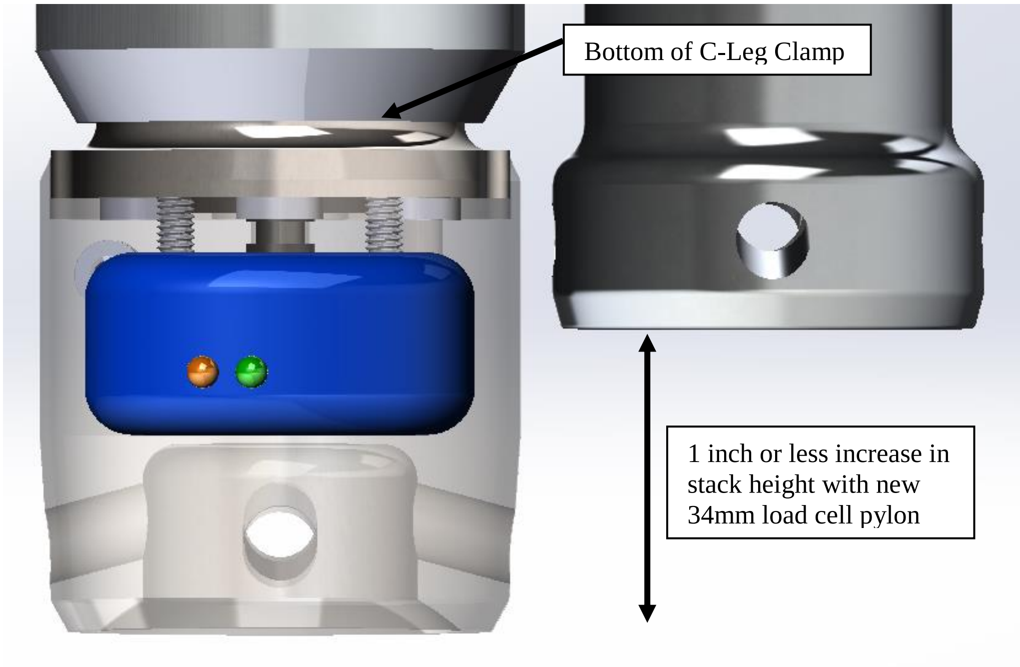
LEGS Open-Source BLE Pylon Load Sensor (V1)



Production of the V1 BLE Pylon Load Sensor

LEGS wireless Bluetooth load sensor, shown on the Odyssey K3 foot - the load cell adds approximately 2in of stack height. It includes an onboard LiPo battery, LiPo safe low-voltage and recharge circuit, charge LED, and power and connectivity LED indicator, along with the standard pyramid connectors

An updated load cell design was made to further reduce the stack height and proceed toward potential commercialization. The shorter load cell pylon utilizes the same components, but replaces the stock 34mm C-Leg pylon with a load cell integrated pylon. This allows the inclusion of all the load cell components while adding less than an inch of stack height.



Compact mechanical design of V2 load cell with 34mm upper post for direct C-Leg integration

Odyssey Ankle Torque Monitoring:

LTI is investigating a commercial effort identify low profile, electronic, threaded, high pressure fluid sensors to indirectly measure the ankle torque on the CPI Odyssey hydraulic ankles. Careful inspection of the specifications is on-going to confirm the best possible solution for this application and it is expected that a final sensor will be selected and ordered early in Year 3 to avoid delays on the project. This will provide ankle torque monitoring integrated with the Odyssey K3 foot.

S7: Software/firmware implementation for gate modification

- All subtasks for Year 2 completed
- <u>Prosthetic hardware control functionality built</u> for implementing conditional output triggering from the arbitrator based on IMU and other sensor data input
- <u>Developed and implemented preliminary conditional statements</u> for knee/ankle coordinated control
- <u>IMU sensor data inputs processed and coordinated</u> across multiple kinematic sensors
- <u>Sensor suite, sensor placement, and data rate requirements fully defined</u> to provide coordinated control of the knee/ankle system
- <u>Coordinated control algorithms under development for knee/ankle</u>, to continue in Year 3: initial development of matrices of sensor inputs and control diagrams for each phase of each task to help refine the ankle control strategies for standing and walking downstairs
- Algorithm coding will be implemented in Year 3 under Task C11

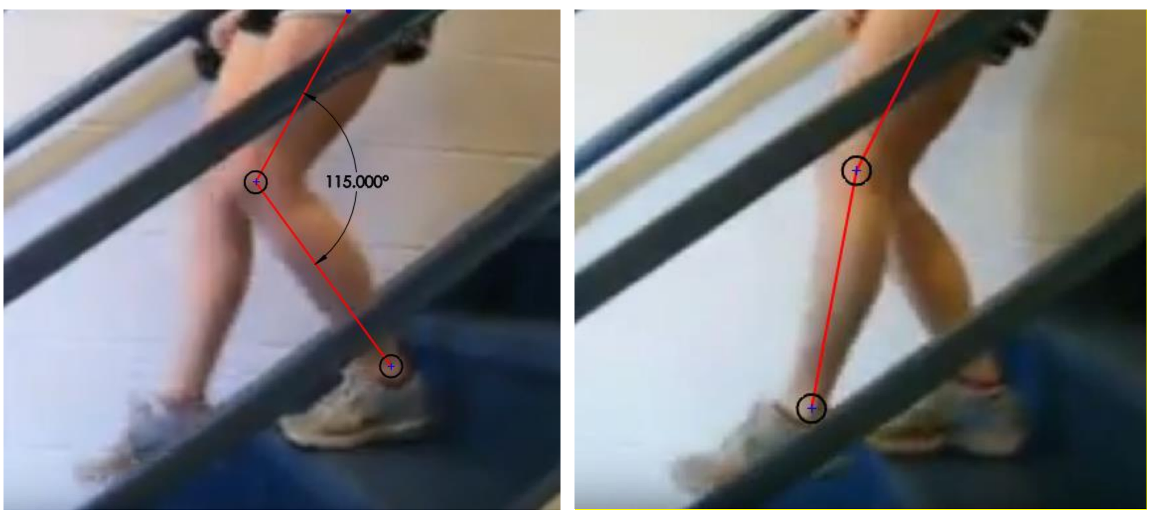
Task S7 Detailed Description

All S7 tasks for Year 2 have been completed. Development of the Scenario 1 gate control algorithm matrix has been completed, and algorithm coding and implementation will be pursued in Year 3 under task C11. To date, a simple sensor-based conditional statement was tested in the

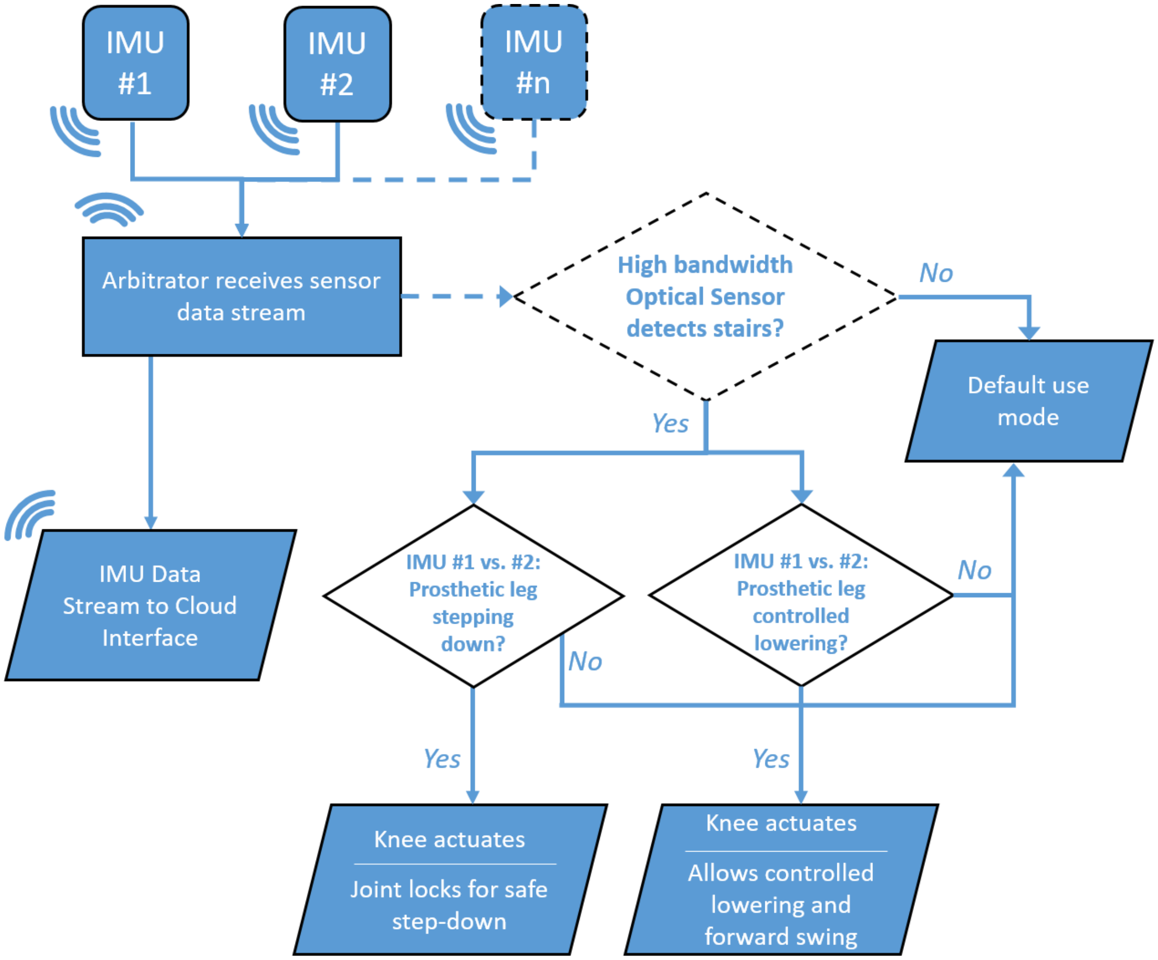
LEGS system, where the arbitrator locks or unlocks the C-Leg knee based on the relative angle of two IMU sensors. This approach will serve as the framework for implementation of the more complex control scenarios outlined in the algorithm matrix.

Output triggering from the arbitrator was implemented, allowing the system to control any electromechanical device in the LEGS environment. The Arbitrator GPIO board supports code written to generate outputs based on sensor inputs and conditional statements. Outputs can include analog, I2C, PWM, SPI, or UART. For the case of controlling knee and ankle damping, which will be demonstrated to aid in general walking and stair descent, the arbitrator wirelessly receives input from the 9-axis IMU sensor and triggers an output based on a conditional statement.

A demonstration scenario was tested with a preliminary coordinated control algorithm. In order for the arbitrator to translate sensor input into an output, conditional tests were implemented at the arbitrator level to determine when an output or state-change is required. Stair descent is being focused on for the initial demonstration and an analysis of able-bodied stair descent was conducted to determine an appropriate conditional statement to control knee damping based on IMU sensor input. Focusing solely on the knee for the time being (knee/ankle coordination is the target), it was determined that a cycle of knee lock and controlled lowering is executed by the leg during stair descent. Knee lock of the descending foot (to support the full weight of the individual) initiates just prior to contact with the stair, as the opposite knee joint is bent ~120 deg or less, and as the descending foot passes through the horizontal plane of the upper foot. Similarly, the upper knee begins bending and controlled lowering just prior to the descending knee locking as the foot swings forward. Using this information, it was possible to utilize IMU data from the able-leg and the prosthetic leg to automatically adjust the damping of the knee joint.



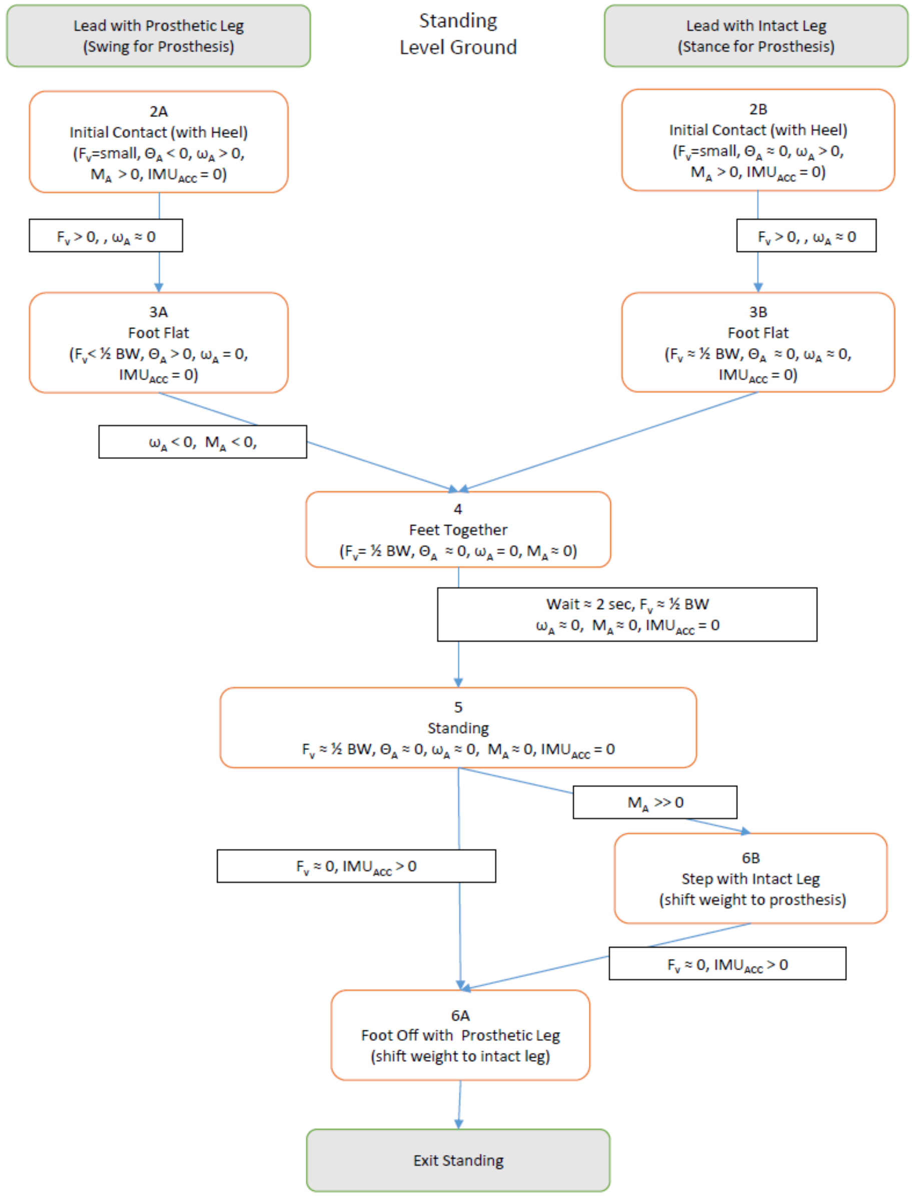
Left: The descending knee (right side) is locked to allow stable support of full body weight as the upper knee bends through ~120 deg. Right: The same (right) knee begins bending and controlled lowering as the left knee is locked and the foot swings forward. Focusing on the prosthetic knee, two states exist: locked and controlled lowering



Logic process of preliminary control algorithm developed in Year 2, focusing on knee control

The two demonstration scenarios, 1. standing on uneven slopes; and 2. walking down stairs, were selected previously; the slope standing scenario was selected to provide a meaningful short term path to implement and demonstrate the hardware and sensor requirements for coordinated prosthetic control. For Scenario 1, sensor input is required to detect the standing state (stair recognition will be accomplished using the video object detection system), and for both scenarios sensor input will allow the developed algorithms to dynamically adjust the prosthetic hardware for the detected scenario.

To accomplish these goals, work this year progressed to define the sensor and data requirements and develop a basic control strategy for these scenarios. Sensor inputs will be based primarily on IMU data input using the completed IMU sensor and the pylon load sensor.



The basic control strategy for standing developed in this performance period – this strategy will be coded and implemented as algorithmic control of the coordinated knee/ankle prosthetic hardware

Under this task the sensor suite was defined for ankle/knee control during standing on slopes and walking down stairs – although the sensor ecosystem will ultimately be plug and play with numerous sensor types and protocols, a suite of demonstration sensors were developed in order to show case coordinated control in the two selected scenarios.

Defining the sensor suite for a robust and thorough algorithmic control strategy proved to be a circular undertaking, where the sensor suite dictates the control strategy and vice-versa. The sensor suite was set to include the IMU sensors mounted on the foot, shin, and upper leg, providing derived upper leg, knee, and ankle angle and rates, along with a wireless load cell built into the lower leg pylon.

LTI completed a thorough exploration of sensors necessary for coordinated control of the two tasks identified for this project. LTI has created matrices of sensor inputs for each phase of each task to help refine the ankle control strategies for standing and walking downstairs previously developed. An example of the matrix for standing with the prosthesis leading in and out of standing is shown below. The goal of these matrices was to confirm the sensors selected for each control strategy, identify any additional sensor options, and potentially address any questions that arise in the development of the current control strategy

Sensor	PIC	PIC-PFF	PFF	PFF-ITO	ITO	ITO-IIC	IIC	IIC-IFF	IFF	Standing	Stand - PF(	PFO
	Event	Transition	Event	Transition	Event	Transition	Event	Transition	Event	wait = 2sec	Transition	Event
Fv	~0	incr.	+ (but small)	incr.	BW	incr., decr, incr.	~BW	decr.	~1/2BW	**constant**		
M_A	~0	decr. (plantar)	-	incr (dorsi)	0	incr.	max +			constant		
M_K	-	- *	-	decr.	-	r, then incr (stay	-			constant		
θ_A	~0	decr. (plantar)	-	incr (dorsi)	-	incr (dorsi)	+			constant		
ω_A	~0	-	0	incr.(	+	incr, then decr	+			**constant**		
θ_K	~0	constant	0	constant	0	constant	0	constant		constant		
ω_K	~0	constant	0	constant	0	constant	0	constant		**constant**		
ACCx	~0	constant	0	constant	0	constant	0	constant		constant		
ACCy	~0	constant	0	constant	0	constant	0	constant		constant		
ACCz	~0	- (but small)	0	constant	0	constant	0	constant		constant		
$roll_{foot}$	~0	constant	0	constant	0	constant	0	constant	0	constant		
$pitch_{foot}$	+	decr. (plantar)	0	constant	0	constant	0	constant	0	constant		
yaw_{foot}	~0	constant	0	constant	0	constant	0	constant	0	constant		
$pitch_{leg}$	-	ıstant or slight iı	-	incr.	-	incr. to positive	~0	constant	0	constant		
	0%		8%		10-12%		50%		58%			

*due to knee being fully extended (0^o) at heel strike. If bent knee, then it would be a initial + moment.

Assumptions:	Ankle +	Dorsiflexion
	Ankle -	Plantarflexion
	Knee +	Flexion
	Knee -	Extension
	Forces and Moments are reported as external	
	x	direction of progression (+ forward)
	y	medial / lateral direction (postive left)
	z	vertical direction (positive up)
	roll	rotation about x axis
	pitch	rotation about y axis (positive up)
	yaw	rotation about z axis

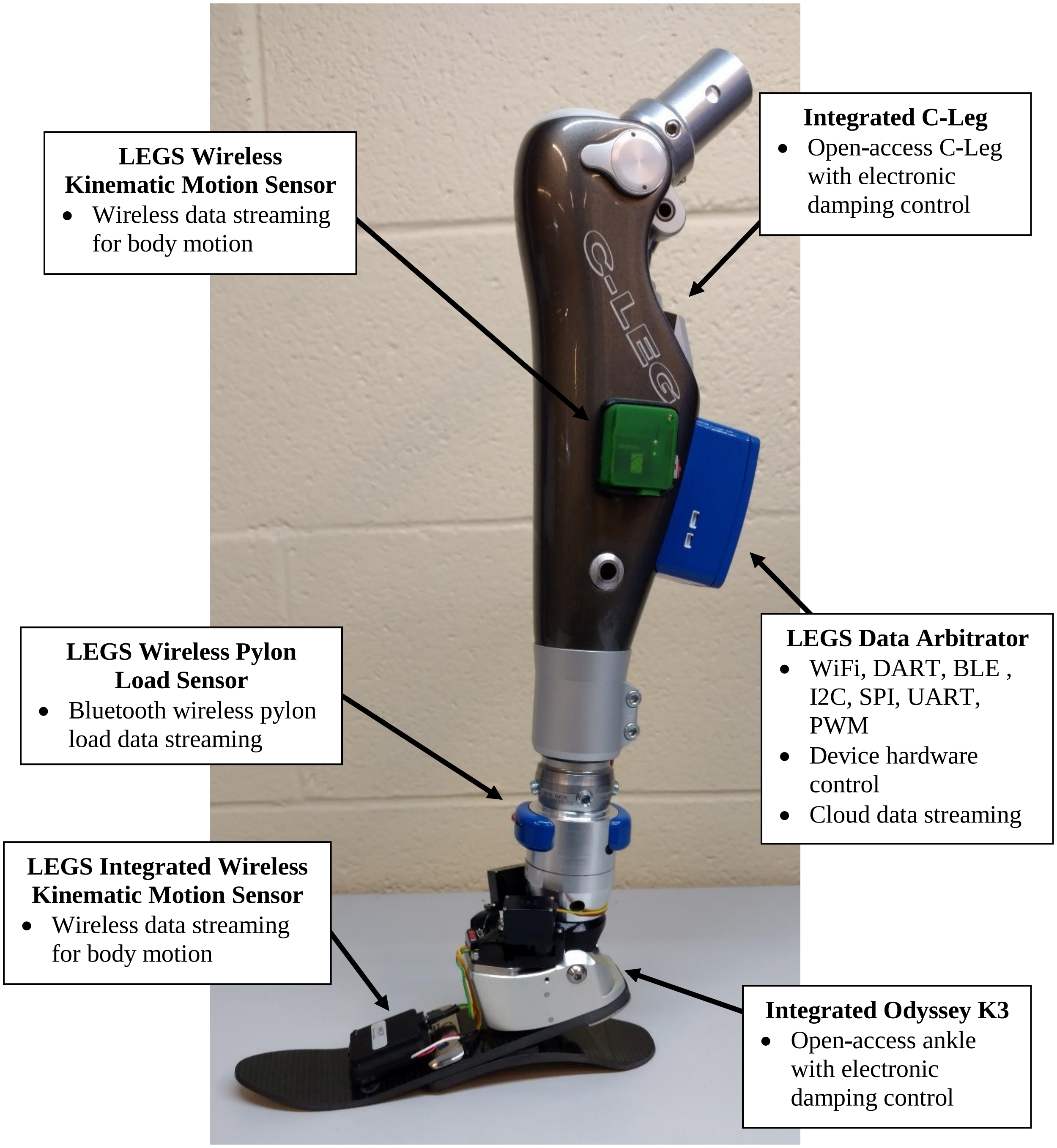
Example of the matrix for standing with the prosthesis leading in and out of standing – these matrices will assist in the development of the coordinated control algorithms in Year 3

S8: Prototype and test integrated sensors for technology demonstration

- All subtasks for Year 2 complete
- <u>Year 2 Prototype</u> includes open electromechanical control of the C-Leg knee and Odyssey K3 ankle, preliminary coordinated control of the C-Leg and Odyssey ankle, integrated wireless IMU and load sensors, live IMU and load sensor data streams and motion views in the LEGS UI, and alerts/feedback demonstration
- <u>Collected IMU data</u> for walking and stair descent scenarios to develop state recognition and coordinated hardware control
- <u>Completed integration of IMU wearable sensor</u> in wristwatch and hardmount format with IMU board, wireless communications, LiPo battery, battery protection circuit with charging passthrough, and integrated battery
- <u>Sensor data streams successfully tested</u> for C-Leg and Odyssey ankle hardware control
- <u>Kinematic IMU sensors tested</u>: sensors successfully stream leg motion data to the cloud, which is then represented by the web-client 3D CGI patient avatar
- <u>Prototype test platform assembled</u> including mounting the LEGS prototype to an adjustable crutch and a hands-free iWalk 2.0 crutch system

Task S8 Detailed Description

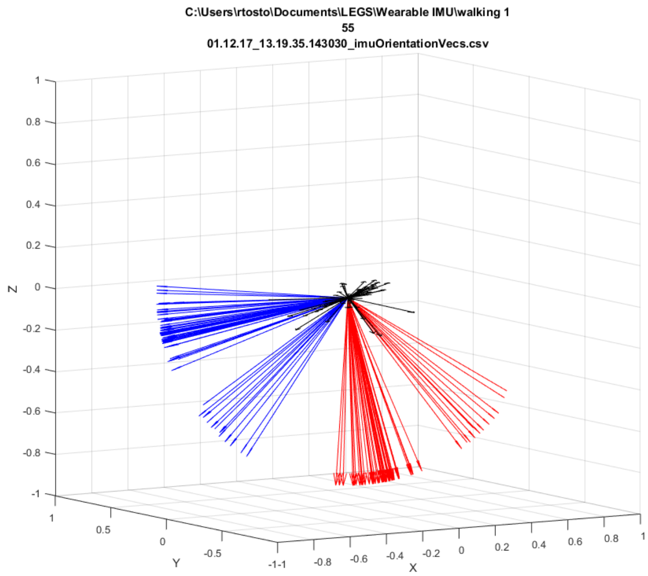
All S8 subtasks for Year 2 have been completed. The prototype LEGS system is shown below, include all of the current sensors, data Arbitrator, integrated C-Leg, and integrated Odyssey foot. The system will be setup for a live demo by the end of October 2017, showcasing the live data streams into the UI, preliminary conditional control of the prosthetic hardware, and functionality of the prototype LEGS system as of the end of Year 2.



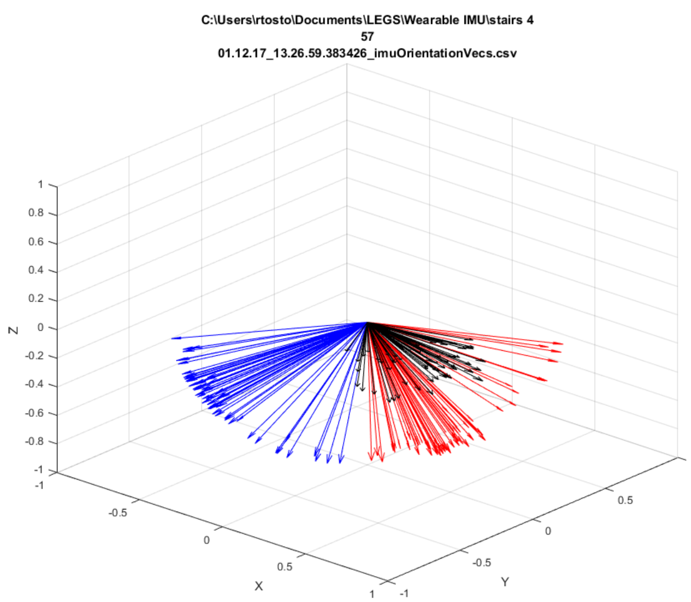
Early in Year 2, initial data streams were collected from the LEGS IMU magnetometer sensor to evaluate patterns in normal walking and stair decent. These preliminary datasets will be used in

Year 3 and coordinate control algorithms are developed to identify the timing of knee/ankle state changes during normal activities.

Data was collected with the LEGS IMU while strapped to the lower leg in two cases: normal walking and stair descent in order to begin developing the conditional coordinated control of the knee/ankle prosthetic. The data plots show the magnetometer orientation and i, j, & k vectors (red, blue, and black, respectively), with the predominant rotation about the k vector, representing the knee. The arbitrator will utilize this data, in addition to the accel and gyro sensors, to determine the relative orientation and motion of the user's limbs and act accordingly to coordinate control of the prosthetic while walking or climbing/descending stairs.

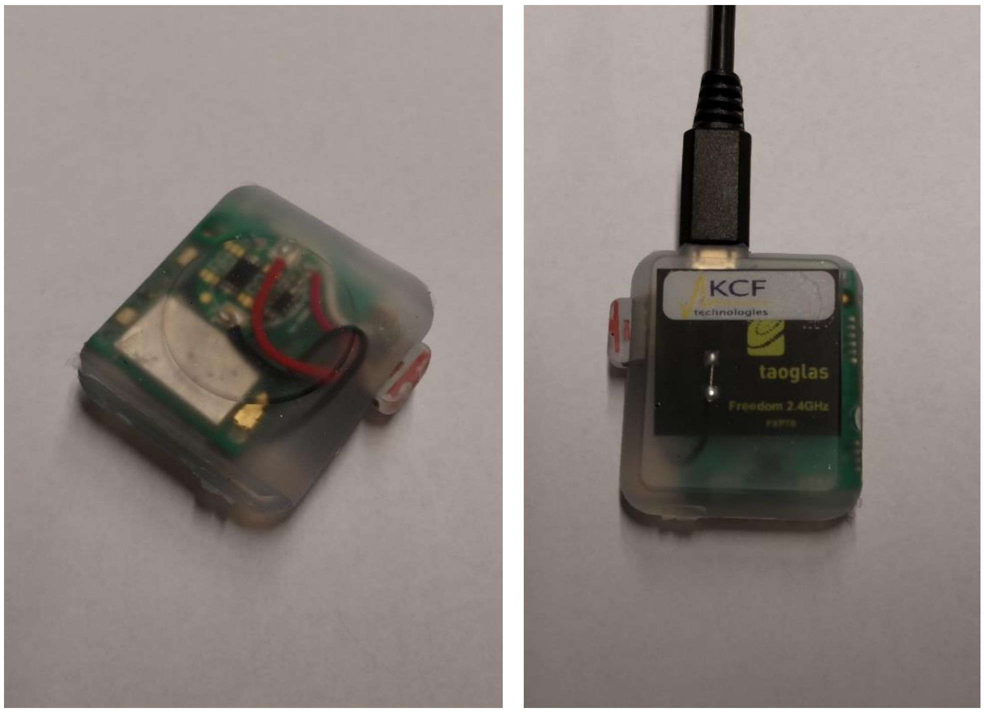


IMU dataset from walking, mounted on the lower leg, showing rotation of the leg below the knee about the k vector (black) axis

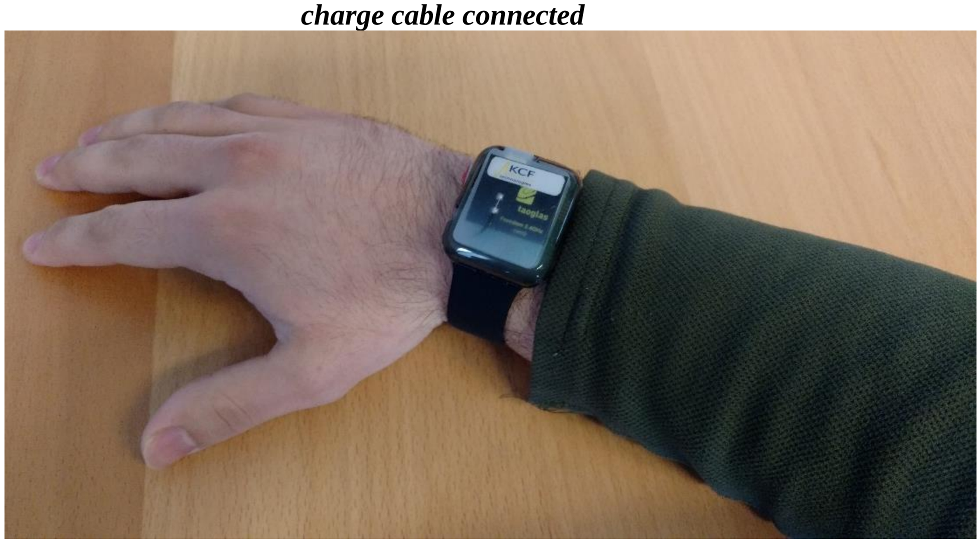


IMU dataset from descending stairs, mounted on the lower leg, showing rotation of the leg below the knee about the k vector (black) axis

IMU sensor integration was completed in Year 2, including wearable IMU watches and hardmount platforms. The IMU sensor and wireless communications board were integrated with a LiPo battery protection circuit, on/off switch, integrated battery, and USB charge port. The protection circuit prevents over-discharge of the LiPo battery and provides a safe charge rate. The sensor overmold is designed to fit with any Apple Watch case or cover and utilize the readily available 3rd party bands and mounts developed for the Apple Watch. The components were overmolded with Smooth-On DragonSkin silicone.



Overmolded KCF IMU wearable sensor – Left: Clear overmolded components, Right: USB charge cable connected



Wearable IMU developed for kinematic monitoring and integration with the LEGS sensor ecosystem

Work this year also included developing a simple but effective method of testing and demonstrating the LEGS system with able bodied volunteers. A straight-forward approach was made by attaching the C-Leg upper knee joint to a modified crutch and an assistive walking device as shown here. The goal being to allow the user to put weight on the LEGS prototype and discern variations in the adaptive control settings while also preventing falls or injury. A single tube adjustable crutch was purchased, along with an iWalk 2.0 hands-free crutch system. Both were modified with a pyramid connector for demonstrating the LEGS prosthetic system in conjunction with the sensor platform being developed.

The prototype test platform was assembled with the iWalk crutch and tested with the lower limb LEGS assembly. The height of the knee support on the crutch prohibits a lower limb assembly exceeding 16inches, making the stack height very short for the adult size C-Leg and prosthetic foot. Using standard hardware, the prototype was assembled and tested and should perform well for LEGS prototype testing.

Testing the LEGS prototype prosthetic assembly

What opportunities for training and professional development has the project provided?

Nothing to report

How were the results disseminated to communities of interest?

KCF has engaged several research groups at the University of Pittsburgh to disseminate the technology into the research community. Several calls and meetings have followed to collect clinical research specifications, and integrate them into the technology development plan. The dissemination of results is a focus of tasking in the third and final year of the project.

What do you plan to do during the next reporting period to accomplish the goals?

During the next year of the project, KCF intends to continue the development process as defined in the SOW. The specific research and development areas are defined by Tasks C9 – C12 and S9 – S12, and the current Gantt chart gives the up to date subtask timeline.

4. IMPACT

What was the impact on the development of the principal discipline(s) of the project?

The full impact of this project will be realized in the last year. During the second year of development, KCF has initiated commercial development of open-interface prosthetic sensor devices, including IMU activity monitors and load cells. In addition, commercialization begun under the DHP13-017 “Advanced Sensor Integration for Prosthetic Socket Monitoring” will transfer to Year 3 work under LEGS and continue in partnership with WillowWood. KCF and LTI have also initiated commercial development of a microprocessor controlled foot.

What was the impact on other disciplines?

Nothing to report

What was the impact on technology transfer?

KCF is maximizing the likelihood of industry adoption by collaborating with industry members and universities, specifically LTI, CoApt, WillowWood, and the University of Pittsburgh. In this period, the focus was on collecting specifications and requirements to define processor and sensor modules that encompass the range of relevant applications. That activity will continue throughout the project as prototype devices enable KCF to elicit feedback that is more detailed.

KCF met with University of Pittsburgh representatives and updated them on the status of the LEGS project, agreeing on a mutual commitment to move into the next phase of teaming. The current status for each group at University of Pittsburgh is:

- Brad Nindl w/ Chris Connaboy – Testing & use of health monitoring wireless sensors for injured soldiers in rehabilitation and injury prevention
    - Status: KCF has delivered & demonstrated prototypes with data streaming to cloud. Chris’ team intends to test in the near future.
- Heather Bansbach w/ Phil Marzolf – Evaluate teaming with AccelMotion (U. Pitt startup) to accelerate their business
    - Status: Have held several meetings. Paused until testing is performed
- Goeran Fiedler – testing and clinical use of Sensorized Socket Liner
    - Status: Testing and collaboration in process. KCF moving ahead to mature the designs and give Goeran prototype units (using Sandia labs sensors, etc.)
- James Irrgang w/ Kevin Bill & Andrew Lynch – Future potential to team with Dr. Irrgang’s research
    - Status: Discussion/sharing of relevant pressure sensor prototypes has occurred
- Brian Vidic – Broader teaming between KCF and Pitt Engineering
    - Status: Jeremy visited in July. Pending follow-up discussion to broaden the scope of teaming

What was the impact on society beyond science and technology?

Nothing to Report

5. CHANGES/PROBLEMS

Changes in approach and reasons for change

KCF is proposing to develop a universal software based, multi-protocol translator service for prosthetic devices to replace PDCP. PDCP was envisioned as a method of achieving CAN-based

universal communications between multiple manufacturer's prosthetic hardware, and was written into the proposal for this project circa 2015. Since the development of PDCP however, few if any prosthetic device manufacturers have investigated using this protocol for numerous reasons, outlined in detail in Section C7.

Based on this information, it is apparent that the use of PDCP has declined and other options should be considered for this project. KCF is proposing to develop a mutli-protocol translator consisting of four major pieces: a sensor data-store; central computing language; coordinated control algorithm-store; and a closed manufacturer protocol library.

This approach would largely address the reasons for non-adoption of PDCP while providing the same outcome envisioned by PDCP. Further information is available as requested.

Actual or anticipated problems or delays and actions or plans to resolve them

Nothing to report

Changes that had a significant impact on expenditures

Nothing to report

Significant changes in use or care of human subjects, vertebrate animals, biohazards, and/or select agents

Not applicable

Significant changes in use or care of human subjects

Not applicable

Significant changes in use or care of vertebrate animals.

Not applicable

Significant changes in use of biohazards and/or select agents

Not applicable

6. PRODUCTS

Publications, conference papers, and presentations

Journal publications.

Nothing to Report

Books or other non-periodical, one-time publications.

Nothing to Report

Other publications, conference papers, and presentations.

Nothing to Report

Website(s) or other Internet site(s)

Nothing to Report

Technologies or techniques

Nothing to Report

Inventions, patent applications, and/or licenses

Nothing to Report

Other Products

KCF intends to investigate marketing: 1) Wearable wireless IMU sensors to our industrial customers for personnel health and activity monitoring; 2) Wireless open-access load sensors for prosthetic and industrial applications; 3) Matured results of the Smart Socket Liner technology in conjunction with WillowWood.

7. PARTICIPANTS & OTHER COLLABORATING ORGANIZATIONS

What individuals have worked on the project?

Name	Company	Title	Hours
Andrew Martin	KCF	Electrical and Software Research Engineer	448
Chris Carl	KCF	Embedded Software Engineer	444
Daniel Warner	KCF	Research Engineer	2
David Kraige	KCF	Mechanical Engineer	40
Jacob Loverich	KCF	Director of Engineering	372
Jeremy Frank	KCF	President, Senior Mechanical Engineer	260.75
Joel Klapper	KCF	Research Engineering Intern	410.5
Joseph LeCouvre	KCF	Software Engineer	886.75
Justin Jacobson	KCF	Software Engineer	210
Lucas Stine	KCF	Software Developer	1027
Mark Edwards	KCF	Software Engineering Manager	336
Ryan Tosto	KCF	Senior Mechanical Engineer	1222
Stephen Wenner	KCF	Electrical Engineer	100
Thomas Tierney	KCF	Machinist	171
Zackary Ridall	KCF	Software Engineer	1849

Personnel with more than one person month (>160 hrs)	
Name:	Andrew Martin
Project Role:	Electrical and Software Research Engineer
Nearest person month worked:	3
Contribution to Project:	Mr. Martin has performed electrical, firmware, software, and UI development on all aspects of this project
Funding Support:	Army Socket Liner 2: W81XWH-14-C-0013
Name:	Chris Carl
Project Role:	Embedded Software Engineer
Nearest person month worked:	3
Contribution to Project:	Mr. Carl has performed work in the area of developing software code for the communication and sensing platform.
Funding Support:	KCF

Name: Project Role: Nearest person month worked: Contribution to Project: Funding Support:	Jacob Loverich Director of Engineering 2 Project requirements definition, development management Army Socket Liner 2: W81XWH-14-C-0013
Name: Project Role: Nearest person month worked: Contribution to Project: Funding Support:	Jeremy Frank Principal Investigator 2 Defining project requirements, architecture; conducting management; partner liaison Army Socket Liner 2: W81XWH-14-C-0013
Name: Project Role: Nearest person month worked: Contribution to Project: Funding Support:	Joel Klapper Research Engineering Intern 3 Mr. Klapper has performed work on the high bandwidth optical sensor system No funding outside award
Name: Project Role: Nearest person month worked: Contribution to Project: Funding Support:	Joseph LeCouvre Software Engineer 6 Software quality testing, validation, and assurance KCF
Name: Project Role: Nearest person month worked: Contribution to Project: Funding Support:	Justin Jacobson Software Engineer 1 Mr. Jacobson has performed work in software development KCF
Name: Project Role: Nearest person month worked: Contribution to Project: Funding Support:	Lucas Stine Software Developer 6 Software code development and data handling Army Socket Liner 2: W81XWH-14-C-0013
Name: Project Role: Nearest person month worked: Contribution to Project: Funding Support:	Mark Edwards Principal Software Engineer 2 Hardware selection and software development No funding outside award

Name: Project Role: Nearest person month worked: Contribution to Project: Funding Support:	Ryan Tosto Senior Mechanical Engineer 8 Mr. Tosto has performed work on mechanical design and production, project management, sensor design and production, testing, and reporting KCF
Name: Project Role: Nearest person month worked: Contribution to Project: Funding Support:	Tom Tierney Machinist 1 Mr. Tierney has machined hardware for housing the sensors and communication modules Army Socket Liner 2: W81XWH-14-C-0013
Name: Project Role: Nearest person month worked: Contribution to Project: Funding Support:	Zackary Ridall Software Engineer 12 Software development for cloud data storage No funding outside award

Has there been a change in the active other support of the PD/PI(s) or senior/key personnel since the last reporting period?

No change

What other organizations were involved as partners?

1. Purdue University
   Location: West Lafayette, IN
   Contribution: Purdue University is working with KCF to define the details of their subcontract deliverables, SOW, and final budget.
2. LTI
   Location: Warren, MI
   Contribution: LTI is working with KCF to define the details of their subcontract deliverables, SOW, and final budget.

8. SPECIAL REPORTING REQUIREMENT: QUAD CHART

Universal Open Power, Communications, and Control for Assistive Devices

DoD Award/USAMRMC Log Number: 14164007
Contract #: W81XWH-15-C-0193

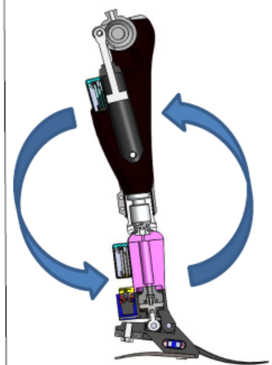
PI: Jeremy Frank **Org:** KCF Technologies, Inc. **Award Amount:** $2,498,732.75

Study/Product Aim(s)

- Advance state of the art prostheses components to support interoperability as well as data and power sharing between devices.
- Develop and demonstrate open-source platform for lower extremity prostheses
- Develop open source communications, flexible energy configuration, advanced high bandwidth sensing, and high energy density actuation technology to be used in prosthetic applications.

Approach

This project will advance the state-of-the-art by addressing the primary technical barriers to achieving the ideal of advanced interoperable prostheses with shared power and data. The approach is to demonstrate the range of technologies that will be required for a range of applications, versus a narrowly focused product development approach developing a single product.



A universal communication, sensor, and power platform enables interoperability between devices, the environment, and the user.

Timeline and Cost

Activities CY	15	16	17	18
Multi-vendor interoperability				
Streaming high-bandwidth data				
Sensor-fusion gait demonstration				
Smart plug-and-play device demo				
Estimated Budget ($K)	**$208k**	**$833k**	**$833k**	**$625k**

Updated: (13 OCT 2017)

Goals/Milestones

CY17 Goals – Component Interoperability
☐Demonstrate multi-vendor prosthetic device interoperability
☐Demonstrate control with streaming high bandwidth sensor data
CY18 Goals – Data Fusion
☐Validate gait control in lab for multi-vendor device interoperability
☐Demonstrate high bandwidth gait control in lab setting
CY19 Goals – Device Integration
☐Smart prosthetic device electronics integration
☐Demonstrate high bandwidth, interoperable smart prosthetic system

Comments/Challenges/Issues/Concerns

- No current challenges, concerns, or issues

Budget Expenditure to Date

Projected Expenditure: $1,665,821 through Sept 2017
Actual Expenditure: $1,377,669.87 through Sept 2017

9. APPENDICES

Appendix A: Coordinated knee/ankle control requirements and guidelines per scenario

- Stair Descent - Ankle coordination with C-leg (C-Leg was selected as an example b/c it has individually variable flexion and extension resistance) may allow the foot to be placed fully on the stair for a more stable base of support, and then stay flat on stair during stair descent as the C-leg is allowing lowering of the body onto the sound limb.
    - There are two primary aspects of stair descent that could benefit from ankle/knee coordination:
        - Loading response: able-bodied individuals (AB) plantar flex while foot by 20-30 degrees when is in the air and then use eccentric dorsiflexion to lower to foot flat. Prosthetic foot is not plantar flexed before contacting stair. If we could detect stair descent, then we could have low plantar flexion resistance to allow for plantar flexion near terminal swing and then ramp up resistance after stair contact to control lowering from the time the toe would hit the stair through foot flat for a smoother transition as weight is transferred off of the sound limb and onto the prosthesis.
        - Controlled lowering: Provide ankle dorsiflexion with knee flexion for smoother transition from prosthetic limb to the sound limb.
            - May need more dorsiflexion range than able-bodied (20-30 degrees) to allow controlled lowering of the sound limb because, unlike able-bodied ankle, there is no articulation at the metatarsal heads and this likely leads to limited heel rise by the prosthetic foot before swing.
    - From a demonstration point of view we'd need to confirm that ankle moment won't trigger "knee break" state in C-leg if the foot is allowed to be fully on the stair. It is likely it will, and thus it would require mods to the C-Leg control system. Therefore, if we try to use this as a demo we'd have to see if the timing OK or, if not, find a way to trick the system.
    - Foot clearance at the start of swing may be more of an issue with the foot not at the edge of the stair (since no active dorsiflexion).
- Jumping Down – for military, the coordination could provide shock absorption during a jump (e.g. down from a truck). After detecting the jump, it would likely be beneficial to correlated the knee and ankle angles with the amount of resistance in these joints (i.e., resistance is relatively low to start but then ramps up as the knee and ankle bend).
    - We could probably detect a jump with sensors in the ankle (ACC, force, etc), but other conditions like going down a fast elevator could trigger the same set of signals and confuse the controller.
    - It would be difficult to know for certain what state the C-leg would be in.
        - If C-leg resistance is so high, knee doesn't bend, ankle we could allow some dorsiflexion and provide some "give" on landing so not so jarring.
        - If C-leg state allows knee to bend, then coordinating ankle response with knee response would provide the most biomimetic motion to off-load the sound limb.
    - VA has drop test machine which could provide nice demo video.
- Running – similar to above
    - Since C-leg enters stance with moderate to high resistance at the knee (though not locked), some "give" at heel strike (with increasing resistance with increasing dorsiflexion angle), could help with shock absorption.
    - If running is detected, the resistance in the knee and/or ankle could be made even higher.
- Slope Walking - During downhill it has been shown that the knee adapts while ankle adapts more when walking uphill (Hansen, 2004).
    - Slope Descent (Declines) – ankle coordination with C-leg would allow for more symmetric gait and more natural knee flexion and improved biomechanics. Generally, having an adaptable ankle helps prevent being propelled down the slope early in stance phase, but coordinating the ankle with the knee should provide greater advantages.

- Ankle/knee coordination could be beneficial if, when going downhill, the ankle would detect the change in slope and then tell the knee to decrease resistance during mid- to terminal-stance.
  - Want to provide relatively low knee flexion resistance to allow the user to 'ride' the knee and prevent 'catapulting' down a slope. This knee flexion will effectively shorten the prosthetic leg length and potentially leads to smaller loads on the sound limb
- Can envision keeping the resistance similar during loading response to allow the ankle to find the ground surface (and thus identifying 'down slope' mode) and then providing a reduced knee flexion resistance during mid to late stance.

- Slope Ascent (Inclines) – ankle coordination with C-leg would allow for more symmetric gait and more natural knee flexion
  - If system detects uphill movement, it would be ideal to limit stance phase knee flexion after loading response to keep the body's CoM as high as possible (i.e., limit the CoM from translating lower and then having the lift the CoM that much higher on the next step).

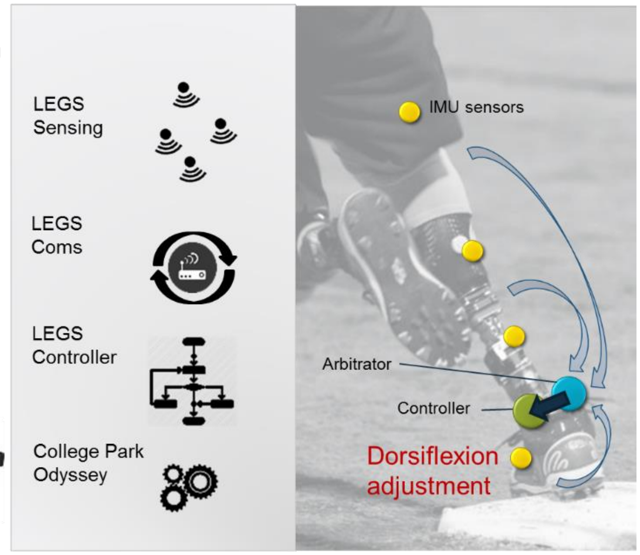
- Level Walking – coordinated control would presumably make walking more symmetric and intuitive for the user.
  - Users who have tried walking with a MP ankle and MP knee that did not have coordinated control indicate it was difficult and took much more concentration than usual.
  - More inputs will help identify gait phase better/more quickly and allow the components to be in sync in order to be more intuitive/less concentration for the user.
  - Swing Phase Dorsiflexion
    - Good for toe clearance
    - Hydracadence knee – exhibits swing phase coupling between knee and ankle for toe clearance (see Sensinger papers)
- Stair Ascent
  - If detected, could set knee extension lower/minimum and knee flexion higher/locked than normal stance phase knee flexion state.
- Squatting – provides better stability than having to balance on toe of the foot when squatting
  - Dorsiflex the ankle once it is identified that both knee and ankle are ultraflexed.
  - Once in position, lock both knee and ankle.
- Standing – more stable / less effort for the user
  - can lock both components when identified that they have been standing.
- In general, shared sensor data could create a more accurate determination of gait state with a greater number of inputs from a greater number of locations.
- Uneven terrain – coordinated control may result in more stability for the user.
- Weighted tasks:
  - Would likely require we control knee resistance since it currently does not respond to weight.
  - Didn't see a dramatic difference in gait with weight during prior work (especially in level walking).
- Climbing a ladder
  - An interesting idea, but this would be a relatively rarely used state that we think is likely to be confused with other more common ADLs with potential negative consequences.

Appendix B: Outcomes

Outcome 1: College Park LEGS MP Foot Ankle

- Overview: Automatically adjust dorsiflexion based on sensor identified usage regime
- Benefit: Reduce falls, improve gait and performance (endurance)
- Outcome: Prototype LEGS compatible microprocessor controlled foot-ankle
- Demonstration: KCF/Army benchtop demonstration

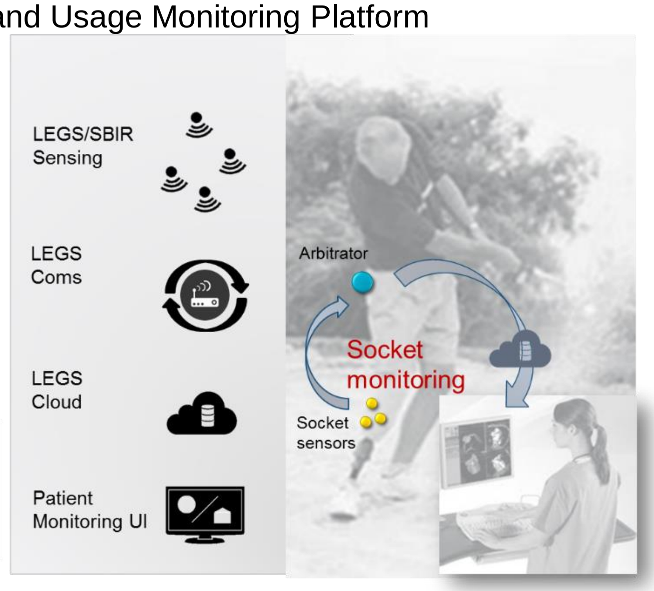
LEGS Compatible MP Foot Ankle



Outcome 2: Socket Health and Usage Monitoring Platform

- Overview: Monitor socket shear, temps, humidity
- Benefit: Detect & reduce undesirable conditions (pressure, etc.)
- Outcome: Prototype Willow Wood Socket
- Demonstration: University of Pittsburgh lab testing (Dr. Fiedler)

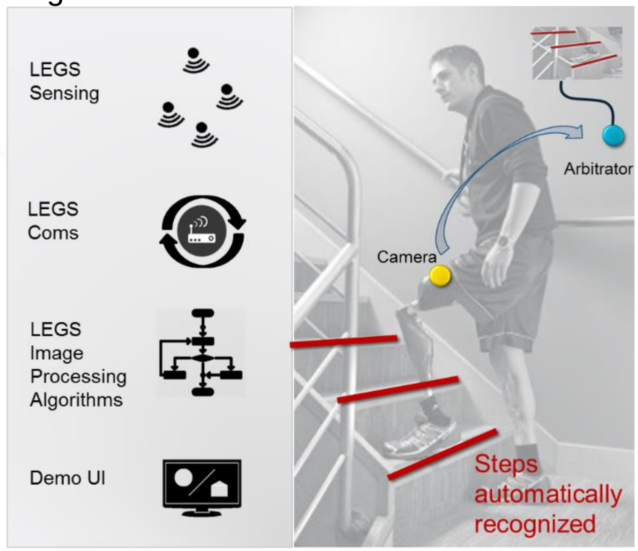
LEGS Socket Health Monitoring



Outcome 3: Visual Object Recognition for Advanced Prosthetics

- Overview: Automatic identification and classification of objects for enhanced prosthetic control
- Benefit: Improve LEGS-enabled performance & avoid falls
- Outcome: Demo platform
- Demonstration: KCF/Army benchtop demonstration

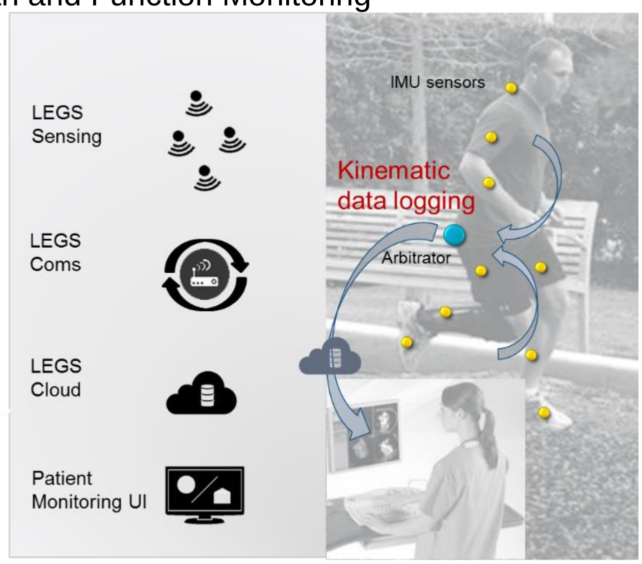
LEGS Object Recognition Demo



Outcome 4: Kinematic Health and Function Monitoring

- Overview: Non-obtrusive enhancement of gait testing & health monitoring
- Benefits: Improve rehabilitation & readiness health assessments
- Outcome: Prototype multi-use LEGS kinematic monitoring system
- Demonstration: University of Pittsburgh lab testing (Dr. Nindl)

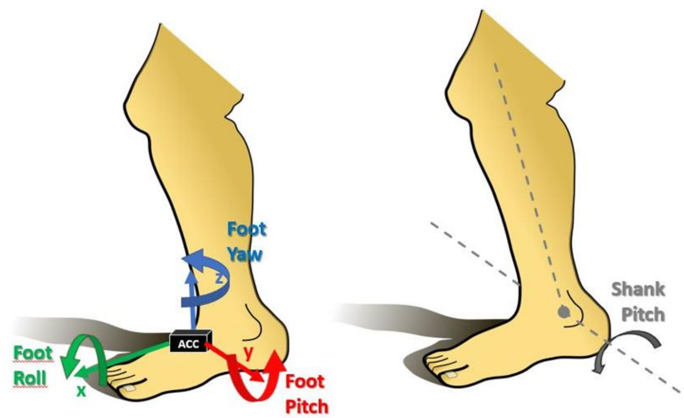
LEGS Kinematic Heath Monitoring Kit



Appendix C: Standing Control Input Matrices

Standing Control Input Matrix

Assumptions:	Ankle +	Dorsiflexion
	Ankle -	Plantarflexion
	Knee +	Flexion
	Knee -	Extension
	Forces and Moments are reported as external	
	x	direction of progression (+ forward)
	y	medial / lateral direction (postive left)
	z	vertical direction (positive up)
Ankle	roll	rotation about x axis
Ankle	pitch	rotation about y axis (positive up)
Ankle	yaw	rotation about z axis
Shank	pitch	rotaton about the y axis
		(postive is leaning towards forward progression)
KEY:	PTO	Prosthetic Toe-Off
	PIC	Prosthetic Initial Contact
	PFF	Prosthetic Foot Flat
	ITO	Intact Toe Off
	IIC	Intact Initial Contact
	IFF	Intact Foot Flat
LEGEND:		potential inputs for transition into/out of standing



Standing - Pros Lead In & Out

Note: values given for Level Walking. Some variables would change depending on slope (e.g. M_A, M_K, Θ_A, ω_A, & $pitch_{foot}$)

Sensor	PIC	PIC-PFF	PFF	PFF-ITO	ITO	ITO-IIC	IIC	IIC-IFF	IFF	Standing		Stand - PF(	PFO	
	Event	Transition	Event	Transition	Event	Transition	Event	Transition	Event	wait = 2sec		Transition	Event	
Fv	+ (but small)	incr.	+ (but small)	incr.	BW	incr., decr, incr.	~BW	decr.	1/2 BW*	constant		decr.	0	
M_A	- (but small)	decr. (plantar)	-	incr (dorsi)	0	incr.	max +	decr.	~0*	constant		constant	0	
M_K	- (but small)**	decr, then incr.	-	incr.	~0	incr, then decr	~0	constant	~0*	constant		constant	small or 0	
Θ_A	+	decr. (plantar)	-	incr (dorsi)	- (but small)	incr (dorsi)	~0	constant	~0*	constant		constant	0	
ω_A	- (but small)	decr, then incr.	0	incr.	+	incr, then decr	~0	constant	~0*	constant		constant	0	
Θ_K	~0	constant	0	constant	0	constant	0	constant	~0*	constant		constant	0	
ω_K	~0	constant	0	constant	0	constant	0	constant	~0*	constant		constant	0	
ACCx	~0	constant	0	constant	0	constant	0	constant	0	constant		incr.	+***	
ACCy	~0	constant	0	constant	0	constant	0	constant	0	constant		constant	~0***	
ACCz	- (but small)	decr, then incr.	0	constant	0	constant	0	constant	0	constant		incr.	+***	
$roll_{foot}$	~0	constant	0	constant	0	constant	0	constant	0	constant		constant	0***	
$pitch_{foot}$	+	decr. (plantar)	0	constant	0	constant	0	constant	0	constant		constant	0***	
yaw_{foot}	~0	constant	0	constant	0	constant	0	constant	0	constant		constant	0***	
$pitch_{leg}$	-	incr.	-	incr.	~0	constant	~0	constant	~0	constant		constant	0	
Gait cycle	*0%*		*8%*		*10-12%*		*50%*		*58%*					

*within a small, but reasonable range due to sway and weight shifting

**due to knee being fully extended (0°) at heel strike. If SPKF or uphill slope, then it could be a initial + moment.

***some small changes may be seen depending on how the foot is off-loaded (e.g. hip hiking, circumduction, etc.)

Standing - Pros In & Intact Out

Note: values given for Level Walking. Some variables would change depending on slope (e.g. M_A, M_K, Θ_A, ω_A, & $pitch_{foot}$)

Sensor	PIC	PIC-PFF	PFF	PFF-ITO	ITO	ITO-IIC	IIC	IIC-IFF	IFF	Standing		Stand - IF(	IFO	IFO-IIC	IIC2	
	Event	Transition	Event	Transition	Event	Transition	Event	Transition	Event	wait = 2sec		Transition	Event	Transition	Event	
Fv	+ (but small)	incr.	+ (but small)	incr.	BW	incr., decr, incr.	~BW	decr.	1/2 BW*	constant		incr.	~BW*	constant?	~BW	
M_A	- (but small)	decr. (plantar)	-	incr (dorsi)	0	incr.	max +	decr.	~0*	constant		constant	+	incr.	+	
M_K	- (but small)**	decr, then incr.	-	incr.	~0	incr, then decr	~0	constant	~0*	constant		constant	~0*	incr.	+	
Θ_A	+	decr. (plantar)	-	incr (dorsi)	- (but small)	incr (dorsi)	~0	constant	~0*	constant		constant	0	constant	0	
ω_A	- (but small)	decr, then incr.	0	incr.	+	incr, then decr	~0	constant	~0*	constant		constant	0	constant	0	
Θ_K	~0	constant	0	constant	0	constant	0	constant	~0*	constant		constant	0	constant	0	
ω_K	~0	constant	0	constant	0	constant	0	constant	~0*	constant		constant	0	constant	0	
ACCx	~0	constant	0	constant	0	constant	0	constant	0	constant		constant	0	constant	0	
ACCy	~0	constant	0	constant	0	constant	0	constant	0	constant		constant	0	constant	0	
ACCz	- (but small)	decr, then incr.	0	constant	0	constant	0	constant	0	constant		constant	0	constant	0	
$roll_{foot}$	~0	constant	0	constant	0	constant	0	constant	0	constant		constant	0	constant	0	
$pitch_{foot}$	+	decr. (plantar)	0	constant	0	constant	0	constant	0	constant		constant	0	constant	0	
yaw_{foot}	~0	constant	0	constant	0	constant	0	constant	0	constant		constant	0	constant	0	
$pitch_{leg}$	-	incr.	-	incr.	~0	constant	~0	constant	~0	constant		constant	0	constant	0	
Gait Cycle	*0%*		*8%*		*10-12%*		*50%*		*58%*							

*within a small, but reasonable range due to sway and weight shifting

**due to knee being fully extended (0°) at heel strike. If SPKF or uphill, then it would be a initial + moment.

***if don't need to release so early, can wait until PTO and use ACCs and zero force as with prosthetic out scenario.

KCF technologies

Standing - Intact Lead In & Out													
Note: values given for Level Walking. Some variables would change depending on slope (e.g. M_A, M_K, Θ_A, ω_A, & $pitch_{foot}$)													
Sensor	**PTO**	**PTO-PIC**	**PIC**	**PIC_PFF**	**PFF**	**Standing**		**Stand - IFC**	**IFO**	**IFO-IIC**	**IIC**		
	Event	**Transition**	**Event**	**Transition**	**Event**	**wait = 2sec**		**Transition**	**Event**	**Transition**	**Event**		
Fv	0	constant	+ (but small)	increase	1/2 BW*	~constant		incr.	~BW*	constant?	~BW		
M_A	+	constant	+ (but small)	incr., then decr	~0*	~constant		constant	~0*	incr.	+		
M_K	+	incr, decr, incr.	- (but small)	constant	~0*	~constant		constant	~0*	incr.	+		
Θ_A	+	constant	+	decrease	~0*	~constant		constant	0	constant	0	D**	
ω_A	0	constant	0	incr., then decr	~0*	~constant	D	constant	0	constant	0	E	
Θ_K	+	incr., then decr.	0	constant	~0*	~constant	E	constant	0	constant	0	K	
ω_K	+	incr, decr., incr.	0	constant	~0*	constant	K	constant	0	constant	0	C	
ACCx	+	incr, decr, incr.	0	constant	0	constant	C	constant	0	constant	0	O	
ACCy	~0	constant	0	constant	0	constant	O	constant	0	constant	0	L	
ACCz	+	incr, decr, incr.	+ (but small)	decrease	0	constant	L	constant	0	constant	0	N	
$roll_{foot}$	0	constant	0	constant	0	constant		constant	0	constant	0	U	
$pitch_{foot}$	-	decr., incr.	+	decrease	0	constant		constant	0	constant	0		
yaw_{foot}	0	constant	0	constant	0	constant		constant	0	constant	0		
$pitch_{leg}$	+	incr. to straight	0	constant	0	~constant		constant	0	incr.	+		
Gait Cycle	*60%*		*100%*										
*within a small, but reasonable range due to sway and weight shifting													
**if don't need to release so early, can wait until PTO and use ACCs and zero force as with prosthetic out scenario.													

Standing - Intact In & Prosth Out										
Note: values given for Level Walking. Some variables would change depending on slope (e.g. M_A, M_K, Θ_A, ω_A, & $pitch_{foot}$)										
Sensor	**PTO**	**PTO-PIC**	**PIC**	**PIC_PFF**	**PFF**	**Standing**		**Stand - PFO**	**PFO**	
	Event	**Transition**	**Event**	**Transition**	**Event**	**wait = 2sec**		**Transition**	**Event**	
Fv	0	constant	+ (but small)	increase	1/2 BW*	~constant		decr.	0	
M_A	+	constant	+ (but small)	incr., then decr	~0*	~constant		constant	0	
M_K	+	incr, decr, incr.	- (but small)	constant	~0*	~constant		constant	small or 0	
Θ_A	+	constant	+	decrease	~0*	~constant		constant	0	D
ω_A	0	constant	0	incr., then decr	~0*	~constant	D	constant	0	E
Θ_K	+	incr., then decr.	0	constant	~0*	~constant	E	constant	0	K
ω_K	+	incr, decr., incr.	0	constant	~0*	constant	K	constant	0	C
ACCx	+	incr, decr, incr.	0	constant	0	constant	C	incr.	+**	O
ACCy	~0	constant	0	constant	0	constant	O	constant	~0**	L
ACCz	+	incr, decr, incr.	+ (but small)	decrease	0	constant	L	incr.	+**	N
$roll_{foot}$	0	constant	0	constant	0	constant		constant	0**	U
$pitch_{foot}$	-	decr., incr.	+	decrease	0	constant		constant	0**	
yaw_{foot}	0	constant	0	constant	0	constant		constant	0**	
$pitch_{leg}$	+	incr. to straight	0	constant	0	~constant		constant	0	
Gait Cycle	*60%*		*100%*							
*within a small, but reasonable range due to sway and weight shifting										
**some small changes may be seen depending on how the foot is off-loaded (e.g. hip hiking, circumduction, etc.)										

Appendix D: Autonomous Stair Detection System

Autonomous Stairs Detection

David J. Cappelleri, Arindam Chowdhury
Purdue University, West Lafayette, IN USA
January 26, 2017

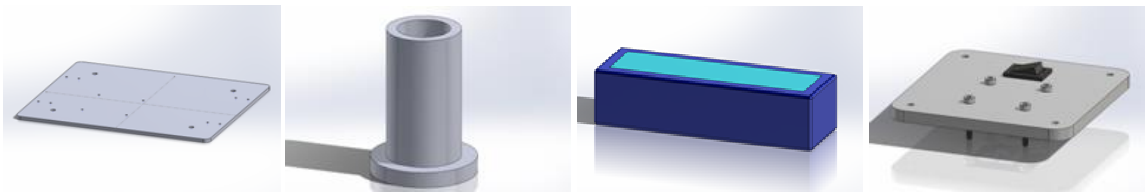
Contents

1. Hardware Setup:

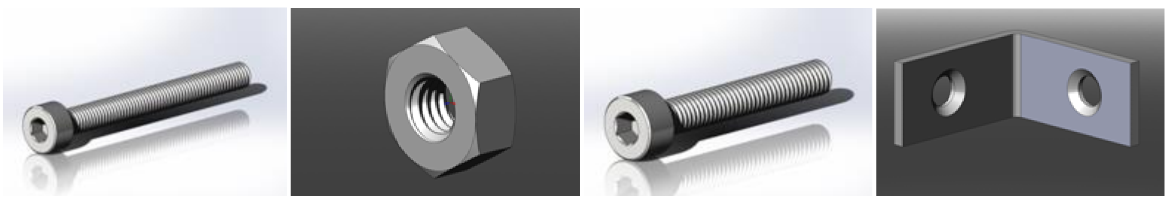
The hardware setup for this project mainly consists of a Microsoft Kinect Depth camera, a computer running the Ubuntu 14.04 operating system, a stand for mounting the camera, and an optional battery for power. The stand has been designed to be manually maneuverable to/from the stairs that are to be detected.

1.1 Hardware components:

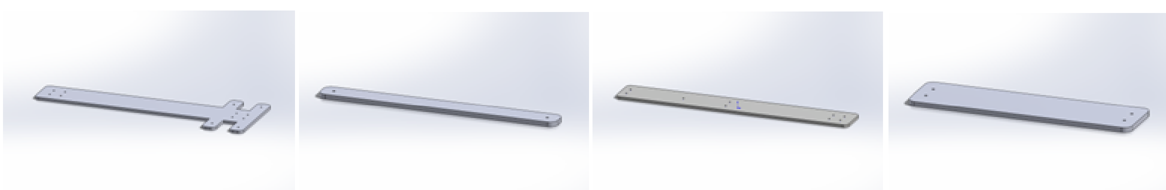
The following diagrams in Fig 1.1 shows the different parts required to assemble the system.



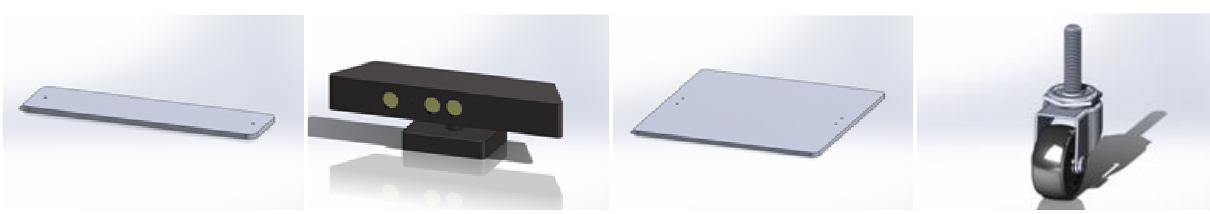
(a) Base (b) Circuit Base Spacer (c) Battery (d) Circuit Base



(e) 30 mm M3 Screw (f) M3 Nut (g) 20 mm M3 Screw (h) Corner Bracket



(i) Part 3 (x2) (j) Part 4 (x2) (k) Part 1 (x2) (l) Part 5



(m) Part 2 (n) Kinect (o) Part 6 (p) Wheel (x4)

Fig 1.1 Hardware components.

The overall hardware setup has two main assemblies – the Lower assembly and the Upper assembly. These two assemblies can be configured in two different ways. One of them is for detecting Model stairs and the other for detecting Real stairs.

In our setup we also show a battery that is used to power the Kinect camera, so that the entire setup can be made mobile. The voltage is supplied from the battery via a voltage regulator that is mounted on the **Circuit Base** along with a Power switch. But in general we can also power the Kinect using its own power adapter connected to a wall power outlet. In that case, we will not using the battery or the voltage regulator and the entire Circuit Base will be bypassed.

1.2 How to assemble the Lower assembly:

The following is the step-by-step procedure to create the lower assembly of the hardware setup.

Take the ***Base*** and mount the four ***Wheels*** to its bottom, as shown in the Fig 1.2. While mounting the *Wheels*, the face of *Base* having the label **'DOWN'** (near one of the corners) should be facing downwards**.** Now, place the four ***Circuit Base Spacers*** on the *Base* in the locations shown in figure. Put the ***Circuit Base*** on top of them and fix it to the Base using four ***30 mm M3 Screws*** and four ***M3 Nuts***. If a battery is used, then fix the ***Battery*** beside the *Circuit Base* (in a convenient location) using the double sided adhesive tape (included in accessories) and connect the positive (red wire) and negative (black wire) of the battery to the positive and negative wires coming out of the *Circuit Base* (the wires are not shown in the figure, but they can be easily noticeable in the actual setup). Now, fix four ***Corner Brackets*** to the *Base* using four *20 mm M3 Screws* and four *M3 Nuts* at the locations shown.

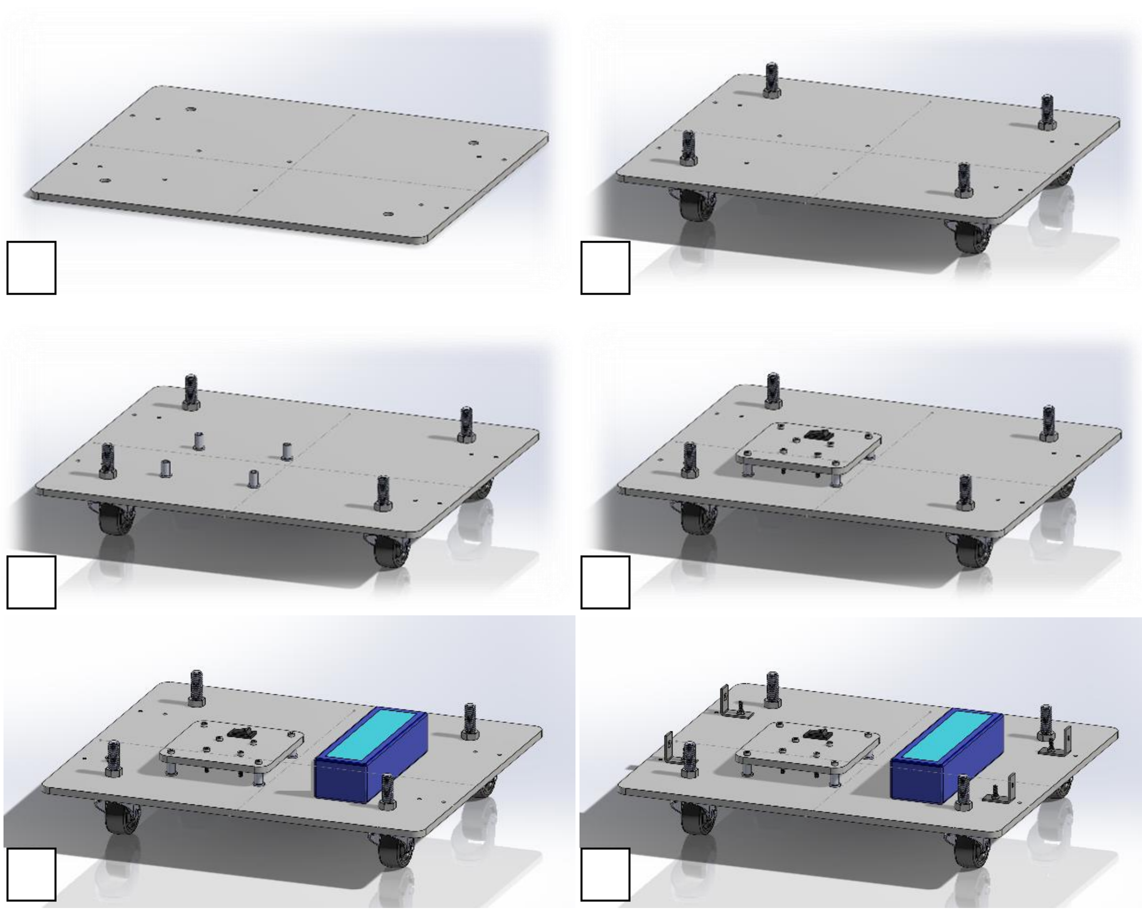


Fig 1.2.

For the next part, refer to Fig 1.3. Fix the two ***Part 3*** pieces to the corner brackets using four *20 mm M3 Screws* and four *M3 Nuts*. Then add one ***Part 4*** using two more *Corner Brackets* to the sides of the *Part 3* pieces (as shown in the figure). Repeat to mount the second **Part 4** to the opposite side of *Part 3*. The Lower assembly is now complete.

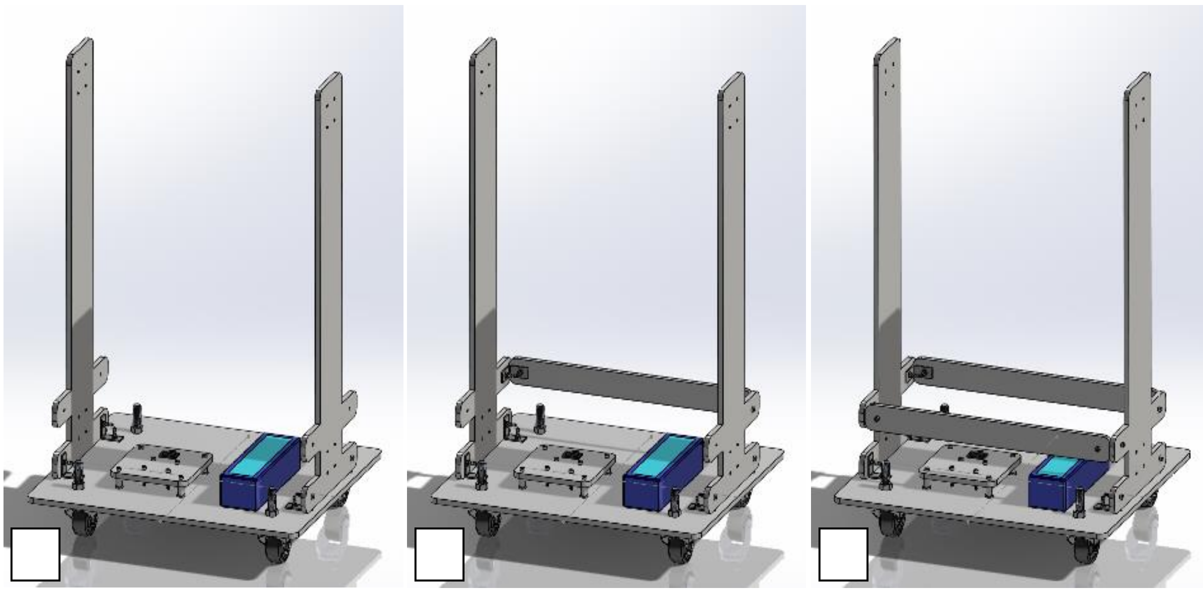


Fig 1.3.

1.3 How to assemble the Upper assembly:

Take the two ***Part 1*** pieces and fix a pair of *Corner Brackets* to each of them, as shown in the Fig 1.4. Join them by using ***Part 5***.

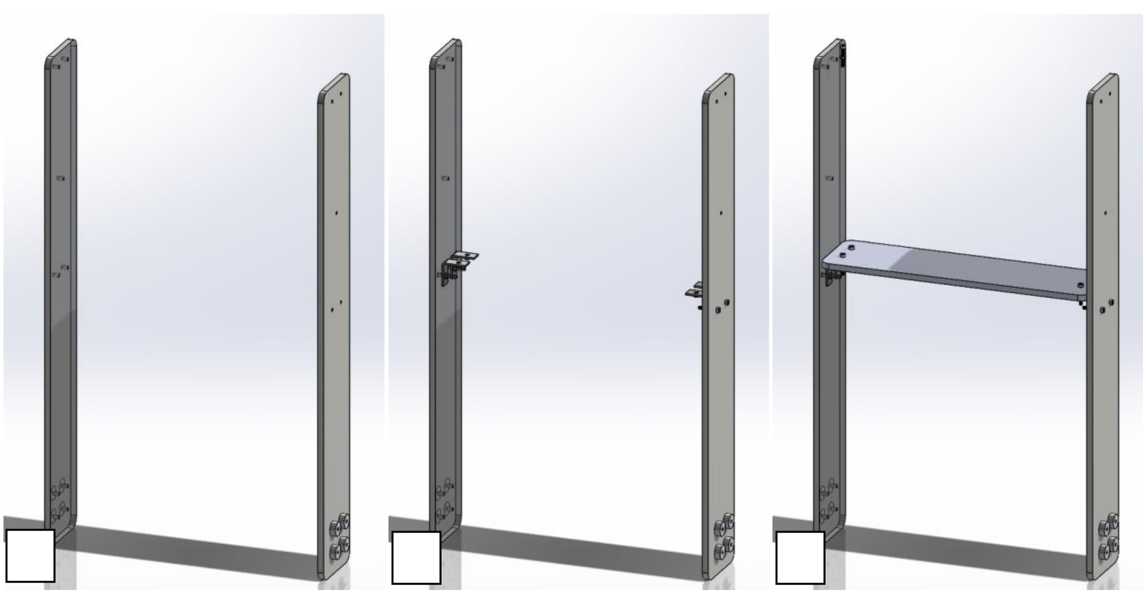


Fig 1.4.

Then mount ***Part 2*** in between them using two more *Corner Brackets* such that it makes and angle of **42° with the horizontal** (as shown in Fig 1.5). Now, fix the ***Kinect*** camera to the *Part 2* using double-sided adhesive tape (included in accessories). Follow the direction shown by the labels (present on the *Part 1* pieces) **'Kinect should face this way'**, while mounting the Kinect. Finally, attach four more *Corner Brackets* to the top portion of each *Part 1* piece and mount ***Part 6*** to these. While mounting the *Part 6*, the face having the label **'DOWN'** (near one of the corners) should be facing downwards. This completes the Upper assembly.

Fig 1.5.

1.4 Assembling the complete setup:

To complete the setup, slide the Upper assembly in between the two *Part 3* pieces of the Lower assembly. They are then fastened together using four *30 mm M3 Screws* and four *M3 Nuts*, near the base of the *Part 3* pieces, on their lateral sides. This is displayed in Fig 1.6.

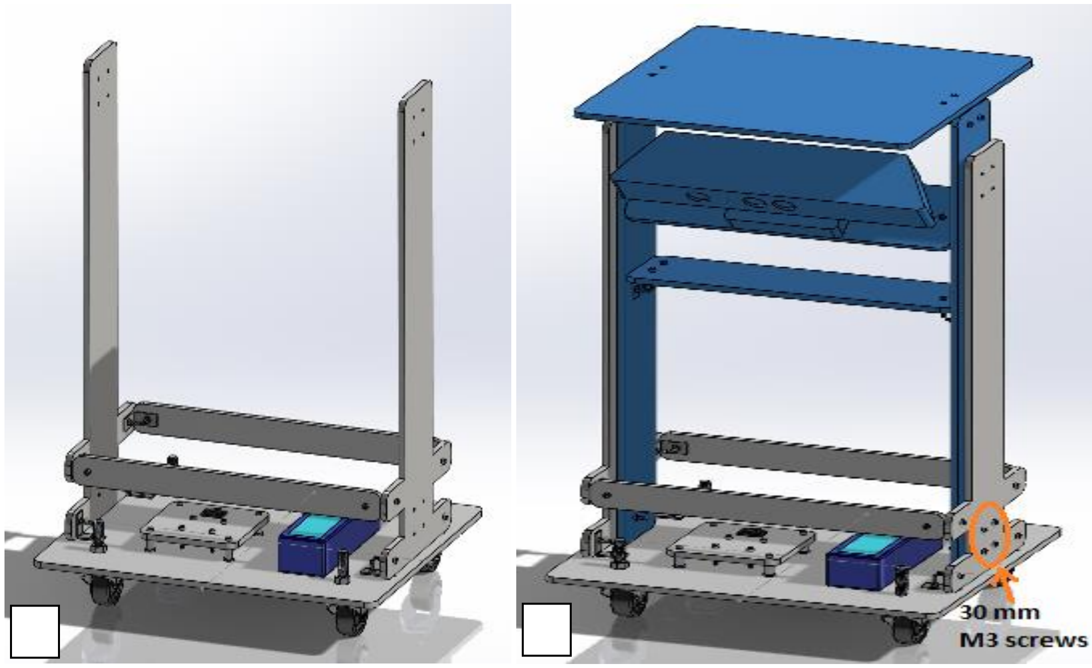


Fig 1.6: The Upper assembly slides into the Lower assembly and they are fastened together using the 30 mm M3 Screws (marked by the orange ellipse), near the base of the Part 3 pieces.

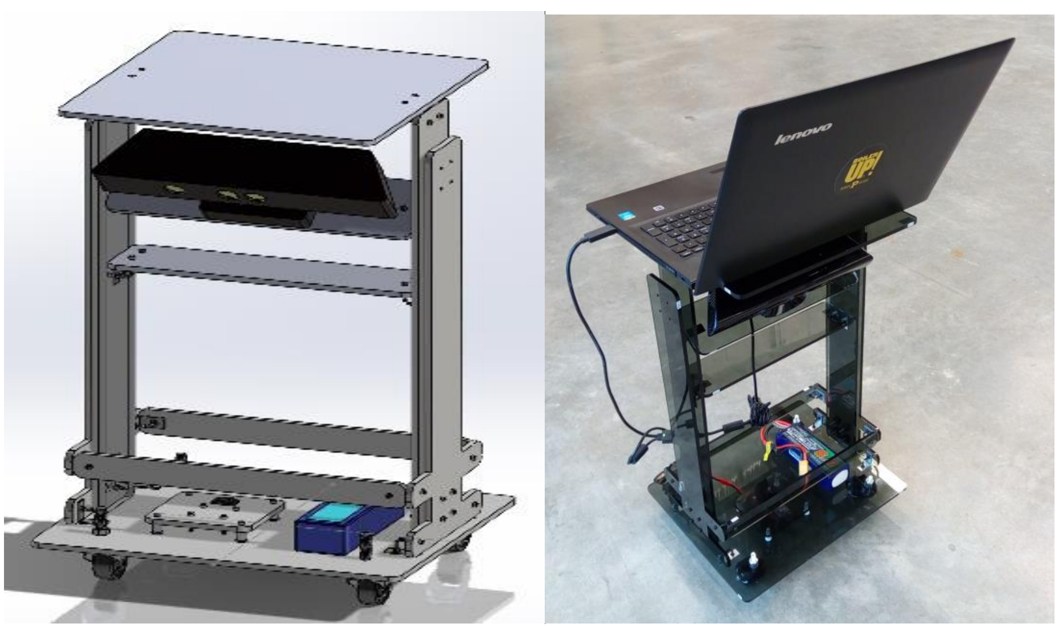
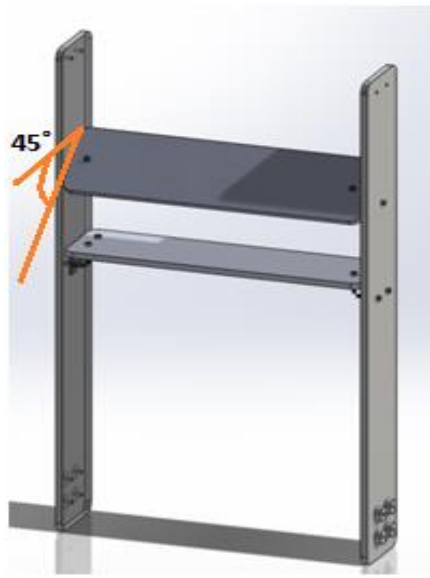


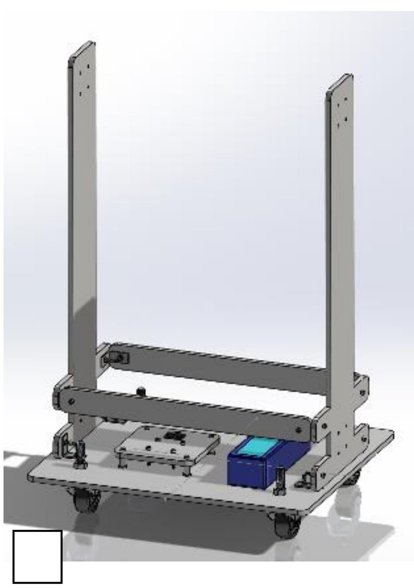
Fig 1.7: The image on the left shows the complete setup. Image on the right shows the actual physical setup for detecting the MODEL stairs.

1.5 Assembling the complete setup for REAL stairs:

The setup for REAL stairs has exactly the same Lower assembly as the previous case. The Upper assembly is also almost the same except that *Part 2* attached in between the two *Part 1* pieces (see Fig 1.5, step 6) is mounted at an angle of **45˚ with the horizontal** (instead of 42˚). This is shown in the following Fig 1.8 (a). Additionally, now the Upper assembly does not mount to the bottom of the Lower assembly. Instead, the it is fastened to the top of the Lower assembly, (using *30 mm M3 screws* and *M3 Nuts*) on their lateral sides (as shown in Fig 1.8 (b) and (c)). The complete assembly is shown in Fig 1.9.



(a)



(b)

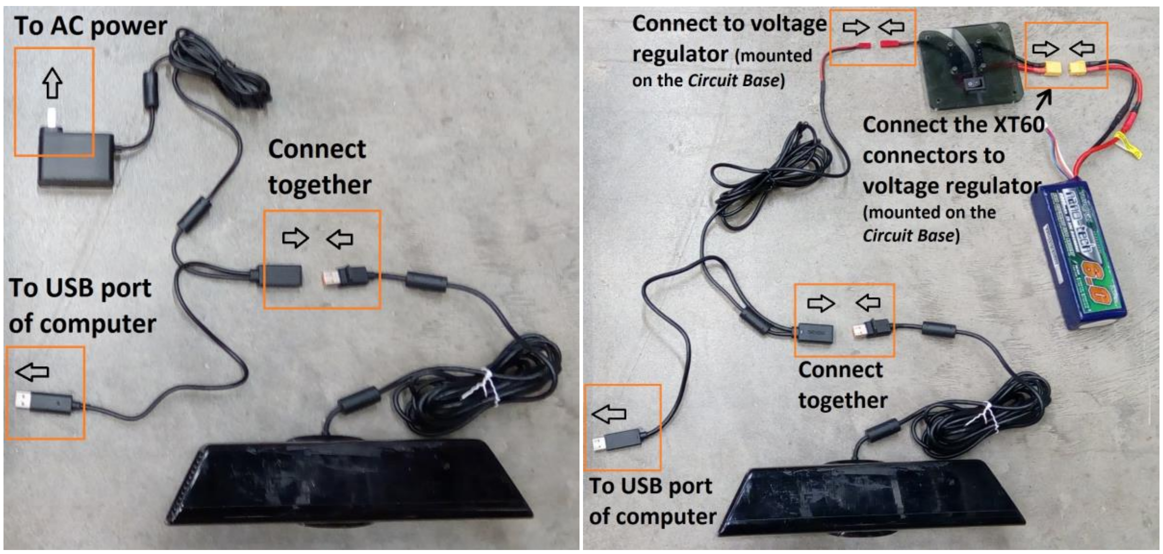
(c)

Fig 1.8 (a): Part 2 is mounted at an angle of 45˚ with the horizontal. (b) Lower assembly for detecting REAL stairs (exactly identical to that of MODEL stairs). (c) The Upper assembly is fastened to the Lower assembly using the 30 mm M3 Screws near the top of the Part 3 pieces.

Fig 1.9: The image on the left shows the complete setup. Image on the right shows the actual physical setup for detecting the REAL stairs.

1.6 Powering up the Kinect:

The Kinect can be powered from an AC power outlet as well as from a battery. The connection of the Kinect using its AC power adapter is shown in Fig 1.10 (a). To power the Kinect with a battery, an ***XT60 battery connector*** should be soldered on to the battery terminals. The positive (red wire) and negative (black wire) of the battery should be soldered to the positive and negative wires of the XT60 connector. This connector is then mated with its counterpart coming out of the voltage regulator (mounted on the *Circuit Base*). Then a separate cable is used to connect the Kinect with this battery. One end of this cable has a red-colored connector (JST 2 Pin Connector), which should be connected to its counterpart coming from the voltage regulator. This connection is shown in Fig 1.10 (b). All the necessary cables are provided in the box for the Kinect.



(a) (b)

Fig 1.10 (a): Powering the Kinect from AC power point (using power adapter). (b) Powering the Kinect using the battery (with XT60 connectors soldered on) through the voltage regulator, mounted on the *Circuit Base.*

The complete list of the hardware components and their corresponding web-links are given in the Appendix.

2. Software Setup:

The software for this project mainly consists of three parts: 1.) the OpenNI and SensorKinect packages, which act as the drivers for the Microsoft Kinect camera; 2.) the OpenCV3 computer vision package, which is used to develop the algorithm for detecting the stairs; and 3.) the main program for stairs detection. All these packages and programs are bundled into the **Stairs_Detection.tar.gz** file. The size of this file is 808 MB and the decompressed version will be approximately 1.2 GB. Therefore, the computer on which this program will run will need to have at least 1.2 GB of free memory space in order to accommodate the entire software package.

The main program in written in **C++ language** and the Linux operating system used to run this code is **Ubuntu 14.04 LTS**.

All the packages used for this project are open source.

2.1 How to install the software:

The step by step process for installing the software is as follows:

1. Use a computer having **Ubuntu 14.04** operating system. Open a terminal window in Ubuntu and go to the directory containing the **Stairs_Detection.tar.gz** file and decompress it by typing the command `sudo tar -xvpzf Stairs_Detection.tar.gz` followed by typing the password of the user when asked for.

This will decompress the file and create a new directory called **Stairs_Detection**. Several messages will be displayed in the terminal while this process is running.

2. Copy and paste the **Stairs_Detection** directory into a convenient location (e.g. home directory or Desktop of the user).
3. Go into the Stairs_Detection directory by typing `cd Stairs_Detection` in the terminal.
4. Inside the directory there is a shell script file called **installation**. Run this script by typing the command `sudo ./installation` in the terminal. Type in the password of the user if asked for.
5. This will install the OpenNI, SensorKinect, and OpenCV3 packages along with their necessary dependencies. Several messages (and maybe some warnings too) will be displayed in the terminal while this installation is going on.
6. This installation will take a long time (might be an hour) as OpenCV3 itself is a very bulky package. But this is a one time operation. Once the installation is completed, an executable file called **Stairs_detecton**, will be created in the same directory.

2.2 How to run the stairs detection program:

1. Plug in the Microsoft Kinect to the computer and wait for some time (may be 10 seconds) to ensure that the computer recognizes the Kinect. The LED on the front end of the Kinect (beside the camera) will light up or blink.
2. Then the **Stairs_detection** executable **file icon** can be **double clicked**, or the command `./Stairs_detection` can be typed in the terminal, to run the program. A display window showing the color and depth video frames will pop up.

2.3 Program controls:

Fig 2.1 shows the display window of the program. The program can run in two modes – the **MODEL stairs detection mode** and the **REAL stairs detection mode**. The current mode is shown near the top right side of the display window. The colored video frames are shown in the screen on the left and the depth video

frames are shown in the screen on the right. By default the program starts up in the MODEL stairs mode. Pressing the '**r**' key on the keyboard switches it into REAL stairs detection mode (as shown in the Fig 2.2). Pressing '**m**' switches the program back to MODEL stairs mode. The distance of the stairs from the camera is shown near the top left side of the display window.

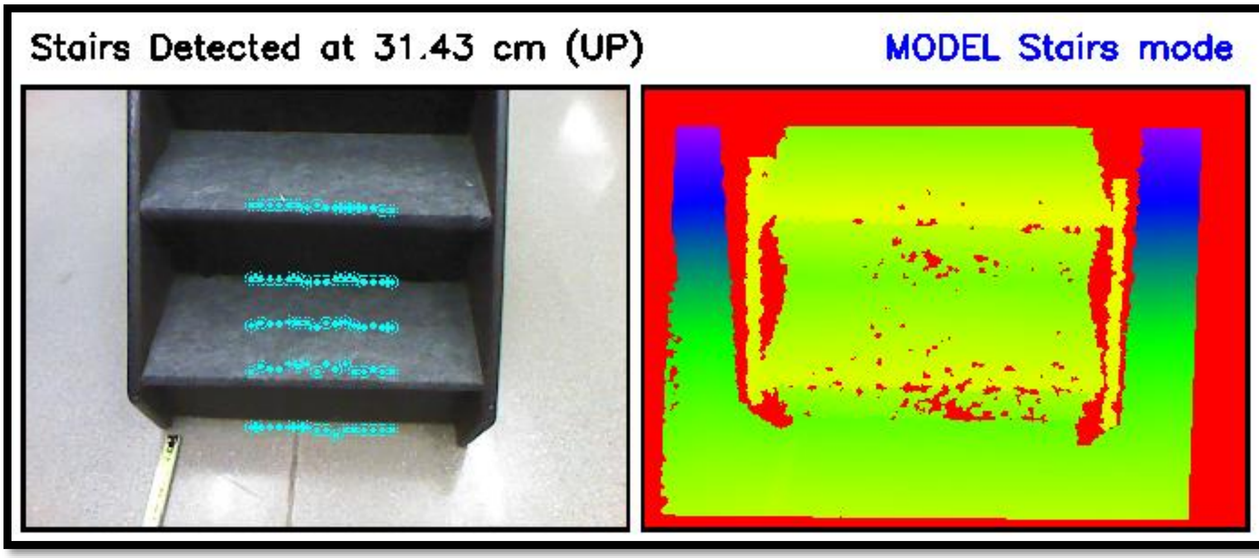


Fig 2.1: Display window showing the color and depth video frames. The program is in MODEL stairs mode and so is able to identify the MODEL stairs. The distance of the stairs and whether it is going UP or DOWN, is also displayed.

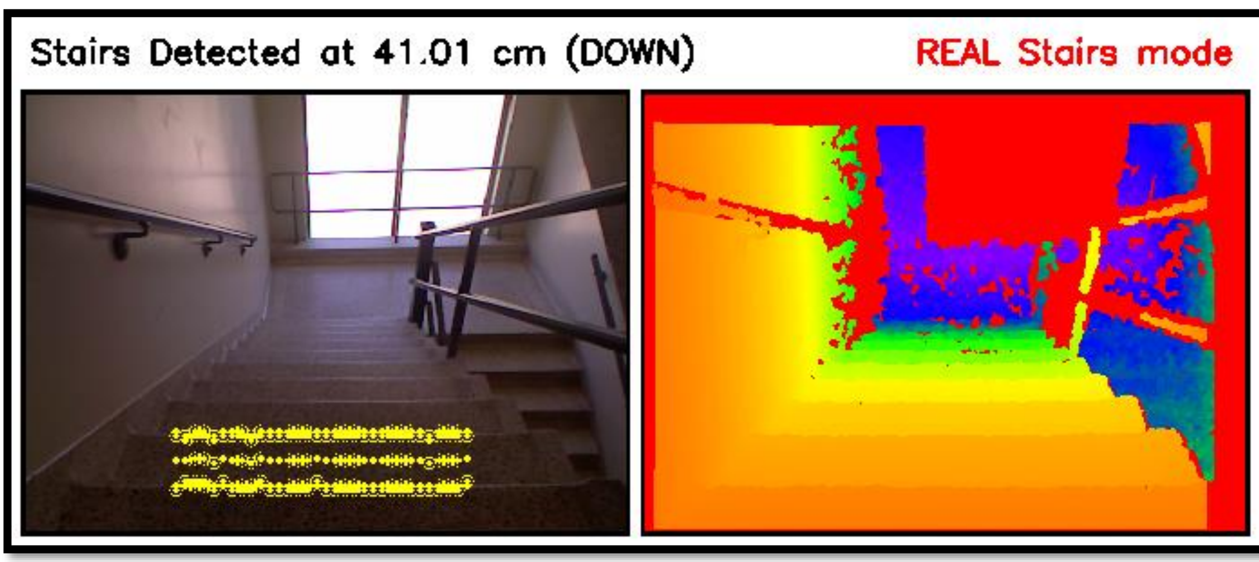


Fig 2.2: The program is in REAL stairs mode and so is not identifying the MODEL stairs

Pressing the '**s**' key saves a copy of the current colored and depth video frames (in the same directory).

Pressing the '**q**' key quits the application and stops the program.

3. Algorithm for Detection of “Down-stairs”:

As we already mentioned, the stairs detection algorithm operates in two modes – the MODEL stairs detection mode and the REAL stairs detection mode. Both these are based on the same principle and work in almost identical manners. The only differences are the values assigned to certain parameters used in the algorithm.

3.1 Assumptions and conventions:

Stairs in general can have multiple steps. However, this algorithm only takes into account the first two steps. This is done for two reasons. First, so that it is able to detect the stairs that actually have only two steps. Second, as a person climbs down the stairs, the number of steps visible to him becomes fewer. And towards the end of the stairs, just before he reaches the ground, only two or three steps might be visible. Naturally the question arises that "why not make the algorithm try to detect only one step then"? Detection of a single step is also possible with this program, but there would be a lot of false detection in that case. This is because there are a number of objects that might resemble a stair in terms of shape. A box lying on the ground, a concrete beam lying in front of the viewer or the edges of the sidewalk beside the streets, etc., all resemble a stair with a single step. Hence, to prevent such false detections, considering only the first two steps of the stairs, seems to be the optimal choice.

As per convention, the positive ‘x’ axis in an image runs from its left to right margin (along the width) and the positive ‘y’ axis runs from its top to bottom margin (along the height). The top left corner of the image is considered as the origin (0, 0).

All measurements in our analysis are in millimeters (mm).

As a convention, OpenCV3 refers to the colored images as BGR (Blue-Green-Red) image. We also follow the same nomenclature in the remainder of this report.

Besides the BGR image, the Kinect camera can also sense the distance of objects within its field of view and show it in the form of an image. This image is referred to as the Depth image. Although this depth image is also represented as a colored image, the colors in it refer to the distance of the objects or points from the camera and not their actual colors in any form. So the depth image can be thought of as a map of the distance of different objects. We will be using the term ‘depth’, to refer to distance in several contexts in this report.

One important point to remember is that the Kinect camera does sense depths of objects less than **two feet** away from it. So our detection algorithm only works when the camera is at a distance more than two feet from the stairs.

3.2 Preprocessing of the images for down-stairs:

There are many other objects that are visible through the camera along with the stairs, like walls next to the stairs, handrails, or any other object. To filter out these unwanted objects, we consider only the lower central part of the image for our analysis. We call this part – the **interest region** of the image, which will be cropped out of the original image. The remaining part of the image is ignored. This interest region is shown in Fig 3.1.

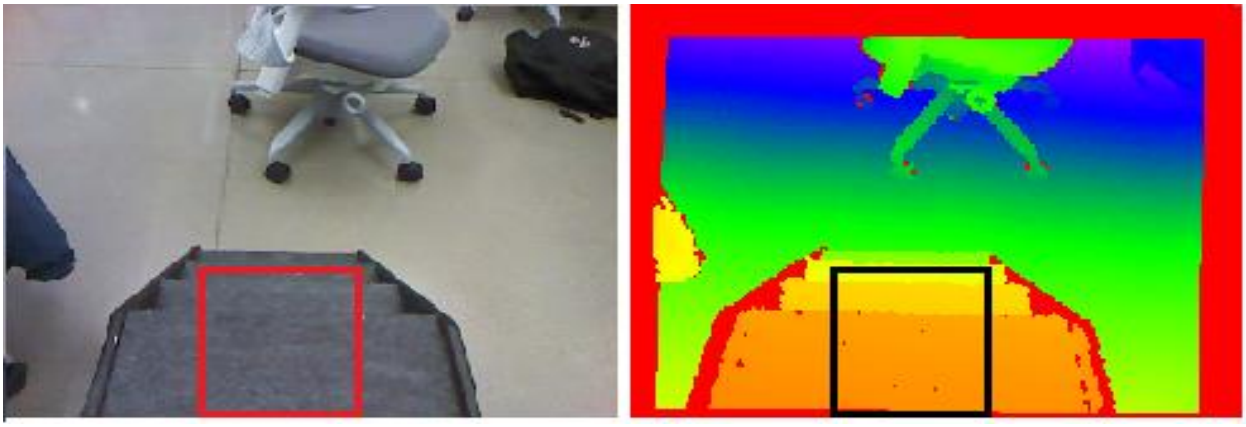


Fig 3.1: BGR and Depth image of MODEL stairs. The Red and Black rectangles show the part of the image used for our analysis - the interest regions. This helps to avoid the unwanted objects captured in the image.

We have specified a certain range of depth (or distance) within which the stairs will be detected, since there is no point in detecting stairs that are too far (like 5 meters) away from the viewer. The Kinect also needs to be mounted at a specified angle and height for most optimized detection of "down-stairs". These preprocessing parameters are listed in the following table.

Parameters for preprocessing: MODEL "down-stairs"	Values
Minimum depth (or distance) below which stairs will not be detected	400 mm
Maximum depth (or distance) above which stairs will not be detected	1200 mm
Angle at which the Kinect is mounted (with the horizontal)	42◦
Height from the ground at which the Kinect is mounted	485 mm
Width of the interest region for down-stairs	128 pixels
Height of the interest region for down-stairs	240 pixels

3.3 Feature Extraction from the images for "down-stairs":

In this step, we will be extracting the key features from the depth image of down-stairs. For this, we are taking multiple parallel scans of the points in the interest region, from its top to the bottom. The **black and red dots** in the depth and BGR images shown in Fig 3.2, shows these scanned points.

If the depths of the scanned points are plotted against the y-coordinates of their corresponding pixels, then we get a plot like the one shown in the Fig 3.3. The plot shows that there is a sudden change in depth of the scanned points at the locations corresponding to the edges of the stairs. The points adjacent to these edges will be our feature points and the abrupt change in the depth will be used to locate and extract these points.

For our algorithm we are considering only the first two steps. Hence, the two points adjacent to the first step, and the two adjacent to the second one, will comprise our set of features for a single scan in the interest region. Therefore, there is a set of four points for each of the scanned lines in the interest region of the image.

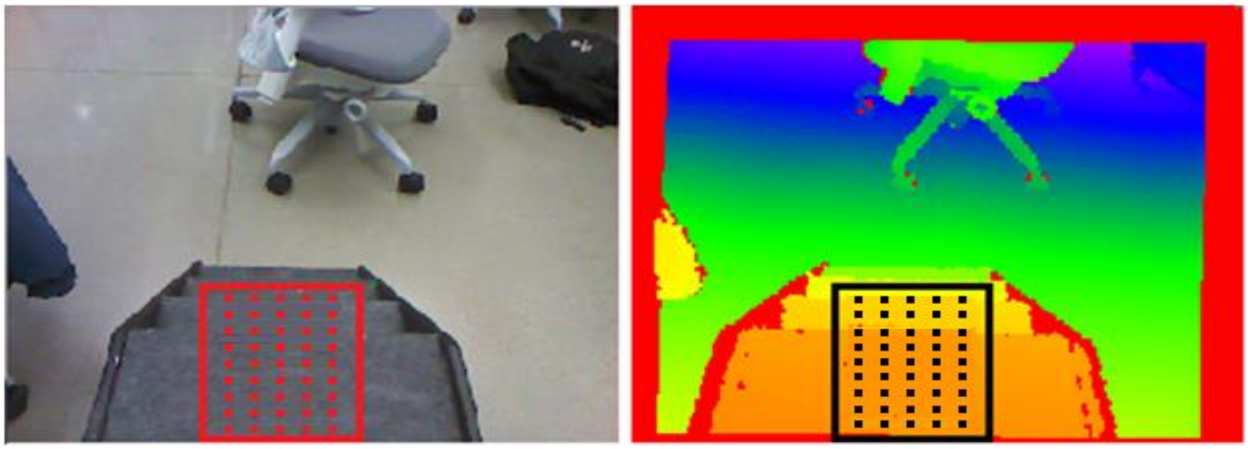


Fig 3.2: BGR and Depth image of MODEL stairs showing the scanned points in red and black dots.

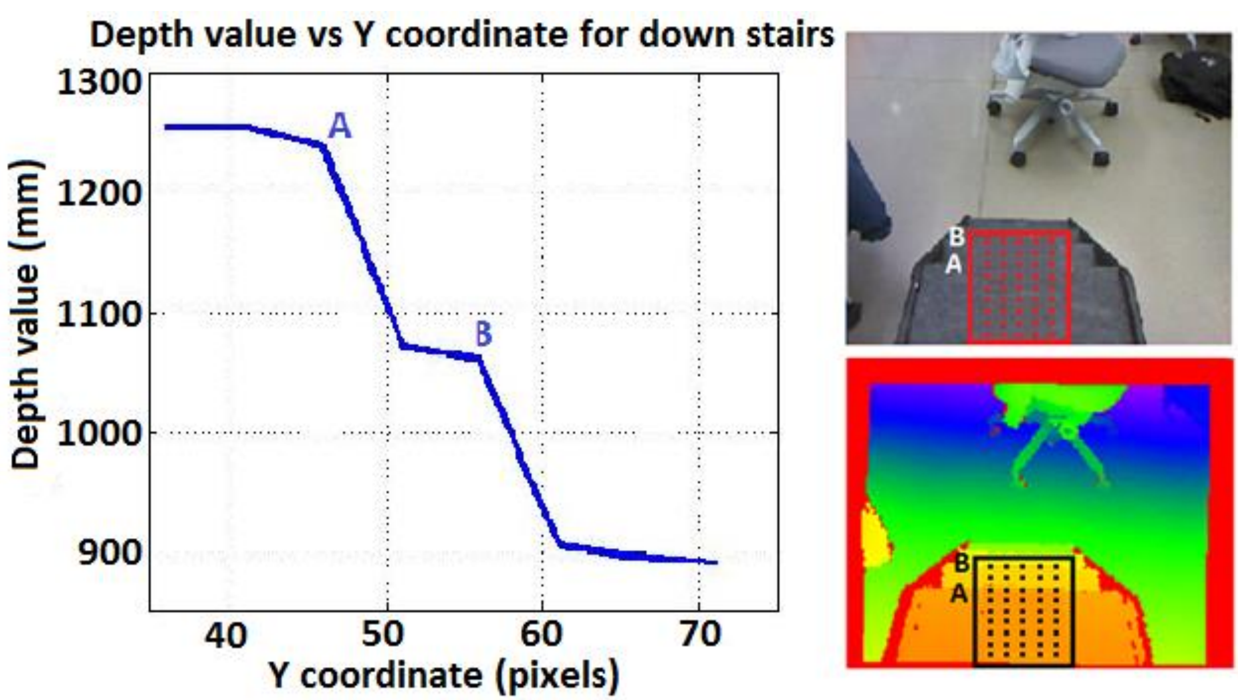


Fig 3.3: Depth of scanned points vs the y-coordinate of their corresponding pixels in the image. The points A and B show the location of the first and the second edges of the stairs. Observe how there is a sudden change in the depth values at these points. This abrupt change in the depth is used to detect and extract these features points.

An example of the four feature points of one particular scan is shown in Fig 3.4.

- P1 = Scanned Point just below the first edge location.
- P2 = Scanned Point just above the first edge location.
- P3 = Scanned Point just below the second edge location.
- P4 = Scanned Point just above the second edge location.

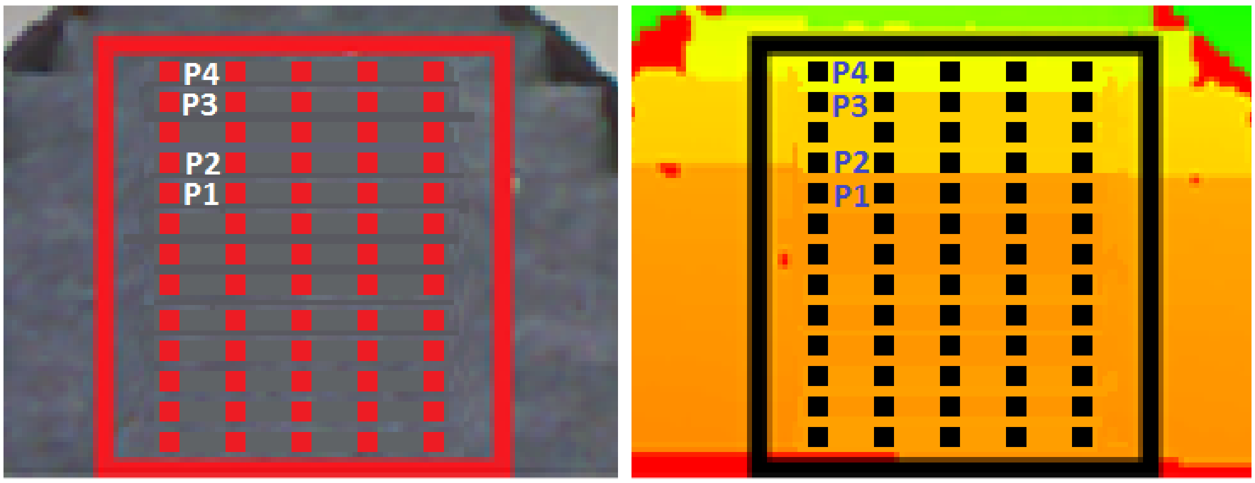


Fig 3.4: Magnified view of the interest region in the BGR and Depth images showing the location of feature points.

Along with this, the average depth of the points between P2 and P3 is also considered as another feature. This average depth represents the depth of the second step of the stairs. This average depth can be considered to be closely equal to the depth of the point midway between P2 and P3.

3.4 Parameterized model of "down-stairs":

Until now we have extracted all the defining features for the stairs. In practice, there might be some other objects in the scene that can also have edges, e.g. the edge of a shelf, chair, set of drawers, etc. In order to know that these features truly represent a "down-stair", we define a set of functions that describes the relationship between these features. This set of functions thus constitutes a parameterized model of the "down-stairs" case.

The functions are as follows:

1. **FUNCTION_1:**

P2.depth = f (P1.y, P1.depth) or,
P2.depth = $\theta_{00} + \theta_{10}$ * P1.y + θ_{20} * P1.depth
--- A function that represents the depth of the P2 in terms of the y-coordinate and depth of P1. θ_{00}, θ_{10}, and θ_{20} are the parameters of this function obtained by linear regression.

2. **FUNCTION_2:**

P3.depth = f (P1.y, P1.depth) or,
P3.depth = $\theta_{01} + \theta_{11}$ * P1.y + θ_{21} * P1.depth
--- A function that represents the depth of the P3 in terms of the y-coordinate and depth of P1. θ_{01}, θ_{11}, and θ_{21} are the parameters of this function obtained by linear regression.

3. FUNCTION_3:

P3.y = f (P1.x, P1.y, P1.depth) or,

P3.y = θ_{02} + θ_{12} * P1.x + θ_{22} * P1.y + θ_{32} * P1.depth

--- A function that represents the y-coordinate of the P3 in terms of the x-coordinate, y-coordinate, and depth of P1. θ_{02}, θ_{12}, θ_{22}, and θ_{32} are the parameters of this function obtained by linear regression.

4. FUNCTION_4:

Average depth of all the points between P2 and P3 is represented by AvD_P2_P3.

AvD_P2_P3 = f (P2.x, P2.y, P2.depth) or,

AvD_P2_P3 = θ_{03} + θ_{13} * P2.x + θ_{23} * P2.y + θ_{33} * P1.depth

--- A function that represents the average depth of all the points between P2 and P3 in terms of the x-coordinate, y-coordinate, and depth of P2. θ_{03}, θ_{13}, θ_{23}, and θ_{33} are the parameters of this function obtained by linear regression.

This parameterized model is created from a set of 53 examples of BGR and depth images of the actual REAL and MODEL "down-stairs" scenario.

3.5 How the algorithm works:

The interest region is first extracted from every frame of the BGR and depth video feed of the Kinect. This region is then scanned to search for feature points. If there are at least two locations along these scans, where the depth changes abruptly, then (assuming them to be potential stair edges) the points adjacent to these locations are extracted as the four feature points (P1, P2, P3, P4). As described in the previous sections, the x and y coordinates and the depths of these points are saved for further analysis. Their values are then plugged into the functions of the parameterized model. Now, the algorithm already knows what the output values of these functions should be if the camera is really looking at the model "down-stairs" case. If we observe that the outputs of the functions are within some close acceptable thresholds of those values, then the algorithm declares that the "model down-stairs" case is in front of the camera. If there was some other object that the camera is looking at, then the functions of the parameterized model will never give proper values all at the same time. This is how the program identifies the "down-stairs". Once a stair is found, the edges are marked, and the distance of the edges from the camera is displayed in the final display window, as shown in Fig 3.5.

The values of the parameters of this model and the accepted thresholds for the function outputs are given in the following table.

Function	Parameters: Model "down-stair"				Function Upper Threshold	Function Lower Threshold
FUNCTION_1	θ_{00} = 164.2443	θ_{10} = -0.2036	θ_{20} = 1.0059		20	-20
FUNCTION_2	θ_{01} = 184.0495	θ_{11} = 0.0413	θ_{21} = 0.9777		20	-20
FUNCTION_3	θ_{02} = -47.0351	θ_{12} = 0.0163	θ_{22} = 0.7540	θ_{32} = 0.075	20	-20
FUNCTION_4	θ_{03} = 13.7039	θ_{13} = -0.0537	θ_{23} = 0.082	θ_{33} = 0.9937	20	-20

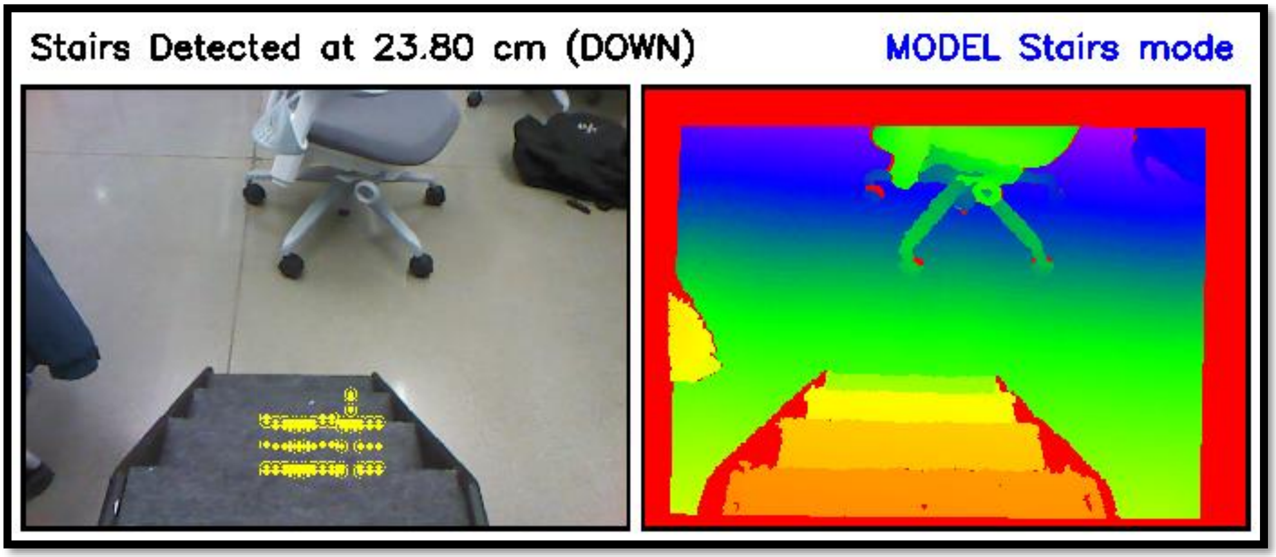


Fig 3.5: Final BGR and Depth images, showing the MODEL down-stairs marked by yellow dots. The window also displays the distance of the stairs from the camera.

3.6 Processing of the images for REAL “down-stairs”:

The algorithm used to detect REAL down-stairs is almost similar to the one used for finding MODEL down-stairs. The only difference is in the values of the parameters used. These parameter values for the REAL stairs are given in the following tables.

Parameters for preprocessing: REAL “down-stairs”	Values
Minimum depth (or distance) below which stairs will not be detected	800 mm
Maximum depth (or distance) above which stairs will not be detected	2100 mm
Angle at which the Kinect is mounted (with the horizontal)	45◦
Height from the ground at which the Kinect is mounted	880 mm
Width of the interest region for down-stairs	320 pixels
Height of the interest region for down-stairs	192 pixels

Function	Parameters: REAL “down-stairs”				Function Upper Threshold	Function Lower Threshold
FUNCTION_1	θ_{00} = 252.523	θ_{10} = -0.3195	θ_{20} = 0.9829		50	-20
FUNCTION_2	θ_{01} = 334.1068	θ_{11} = -0.0223	θ_{21} = 0.985		30	-20
FUNCTION_3	θ_{02} = -85.0389	θ_{12} = -0.0056	θ_{22} = 0.7835	θ_{32} = 0.041	30	-20
FUNCTION_4	θ_{03} = 40.4137	θ_{13} = -0.0064	θ_{23} = 0.1426	θ_{33} = 1.001	20	-30

The following figure shows the detection of REAL down-stairs.

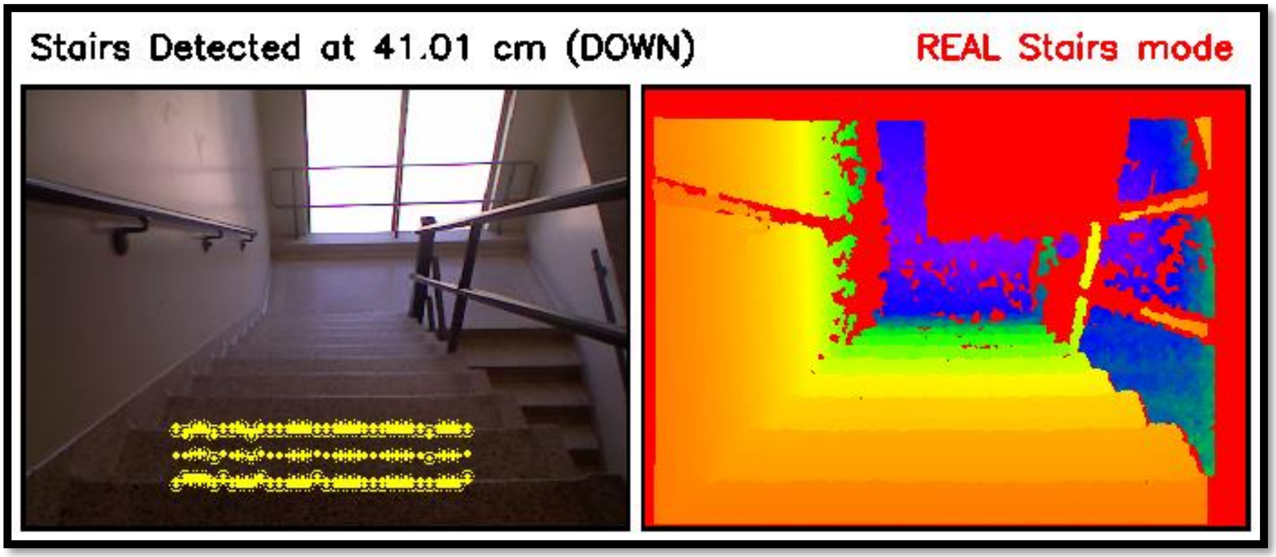


Fig 3.6: Final BGR and Depth images showing the detection of REAL down-stairs.

4. Algorithm for Detection of “Up-stairs”:

The up-stairs detection algorithm also has two modes – one for detecting MODEL stairs and the other for detecting REAL stairs.

4.1 Assumptions and Conventions:

Here also we take into account only the first two steps of the stairs, for the same reasons as discussed in the case of down-stairs.

4.2 Preprocessing of the images for: MODEL “up-stairs”:

We define an **interest region** here for filtering out the unwanted objects like walls and handrails, or any other objects. This region is shown in Fig 4.1.

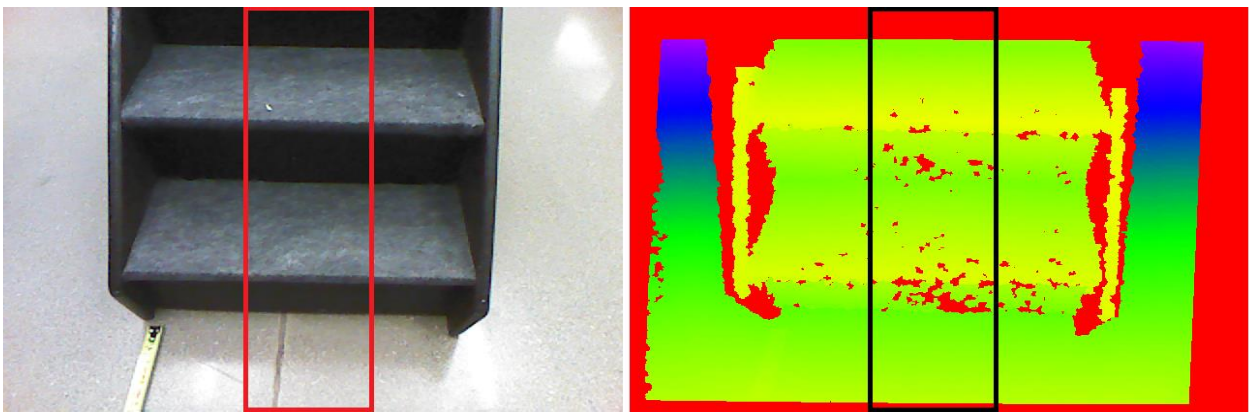


Fig 4.1: BGR and Depth image of MODEL stairs. The Red and Black rectangles show the part of the image used for our analysis - the interest regions. This helps to avoid the unwanted objects captured in the image.

The necessary parameters to preprocess the images for detection of up-stairs, are given in the following table.

Parameters for preprocessing: Model “up-stairs”	**Values**
Minimum depth (or distance) below which stairs will not be detected	400 mm
Maximum depth (or distance) above which stairs will not be detected	800 mm
Angle at which the Kinect is mounted (with the horizontal)	42◦
Height from the ground at which the Kinect is mounted	485 mm
Width of the interest region for down-stairs	160 pixels
Height of the interest region for down-stairs	480 pixels

4.3 Feature Extraction from the images for up-stairs:

In this step, we will be extracting the key features from the depth image of up-stairs. For this, we are taking multiple parallel scans of the points in the interest region, from the top to the bottom (just like we did in the case of *down-stairs*). The **black and red dots** in the depth and BGR images shown in Fig 4.2, shows these scanned points.

If the depths of the scanned points are plotted against the y-coordinates of their corresponding pixels, then we will get a plot like the one shown in the Fig 4.3.

The plot shows that there is a change in the slope of the graph at the locations corresponding to the inner edges at the base of the steps and also at their outer edges. At each of the inner edges the graph hits a local **maxima**, where the slope changes from positive to negative, and at each outer edge, there is a local **minima**, where the slope changes from negative to positive. These maxima and minima points will be our feature points and the change in the slope of the graph is used to locate and extract these points.

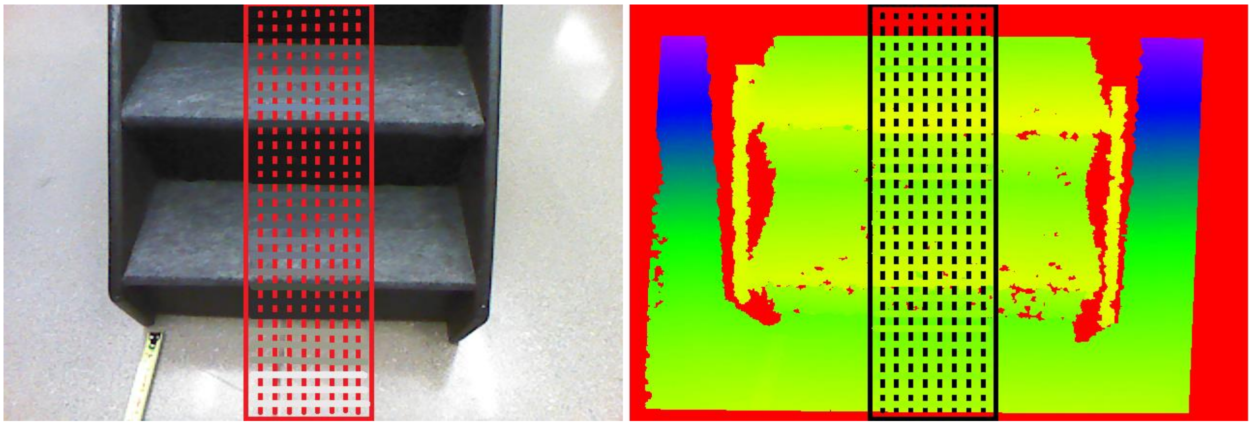


Fig 4.2: BGR and Depth image of MODEL stairs showing the scanned points in red and black dots.

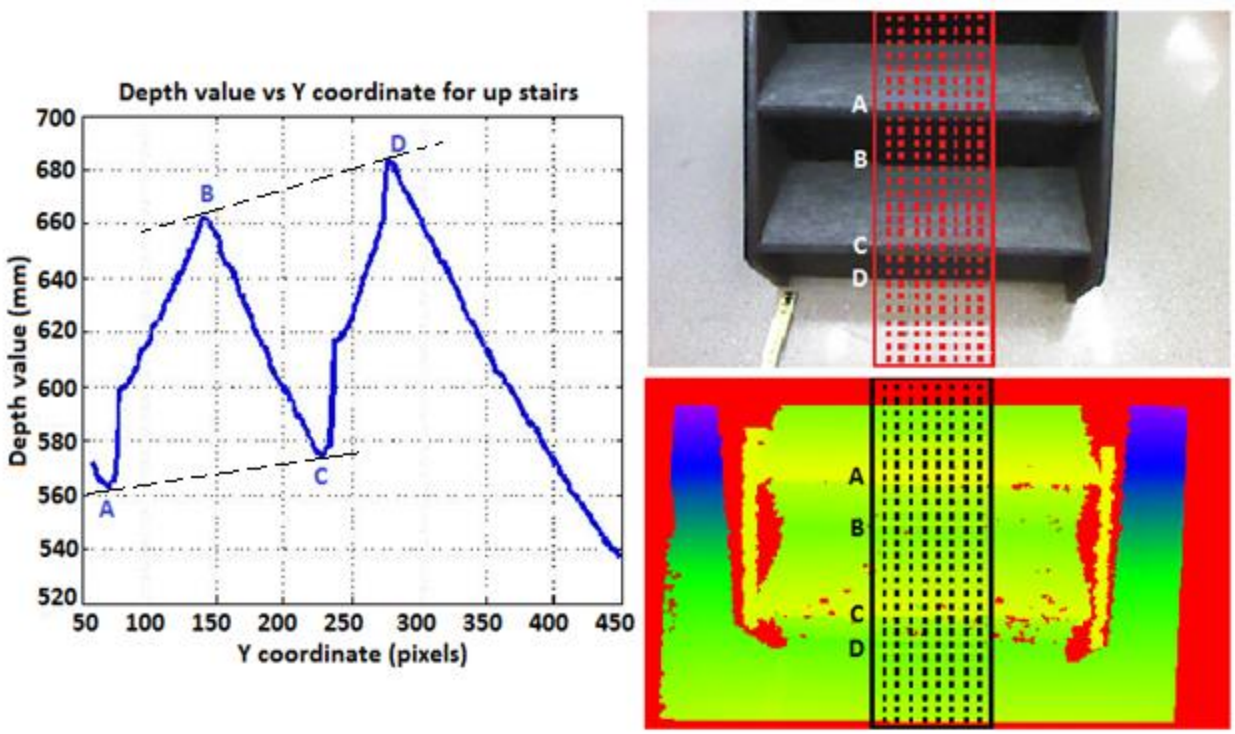


Fig 4.3: Depth of scanned points vs the y-coordinate of their corresponding pixels in the image. The points D and B show the maxima points, which are the inner edges at the base of the first and the second step respectively. C and A show the minima points, which are the outer edges of the

first and the second step respectively. Observe how the slopes change at these locations. These changes in the slopes are used to detect these features points. The slope of the lines CA and DB will also be considered as features.

An example of the four feature points of one particular scan is shown in Fig 4.4.

- P1 = Scanned Point on the inner edge at the base of the first step (first maxima).
- P2 = Scanned Point on the outer edge of the first step (first minima).
- P3 = Scanned Point on the inner edge at the base of the second step (second maxima).
- P4 = Scanned Point on the outer edge of the second step (second minima).

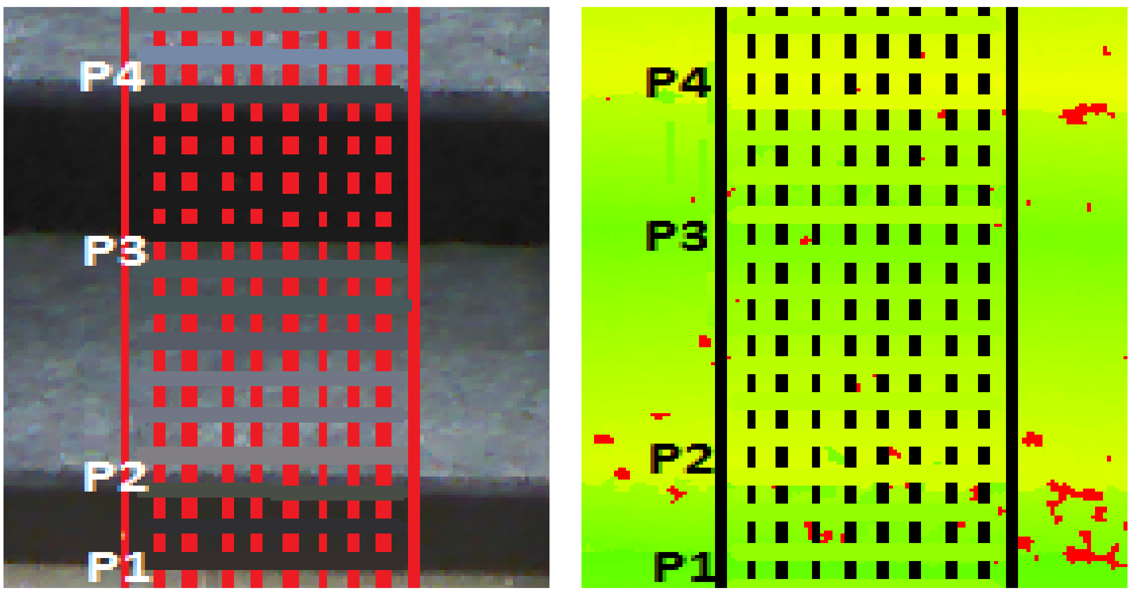


Fig 4.4: Magnified view of the part of the interest region in the BGR and Depth images showing the location of feature points.

Along with this, the average depth of the points between P2 and P3 is also considered as another feature. This average depth represents the depth of the first step of the stairs. This depth can be considered to be closely equal to the depth of the point midway between P2 and P3.

The slope of the virtual line segment joining the points P1 and P3, and the segment between the points P2 and P4 are also considered as features. These are shown as the lines DB and CA in Fig 4.3.

4.4 Parameterized model of up-stairs:

Similar to the case of the "down-stairs", in order to know that these features truly represent "up-stairs", we define a set of functions that describes the relationship between these features. This set of function thus constitutes a parameterized model of the Model "up-stairs" case.

The functions are as follows:

1. **FUNCTION_1:**

P2.depth = f (P1.y, P1.depth) or,
P2.depth = $\phi_{00} + \phi_{10}$ * P1.y + ϕ_{20} * P1.depth

--- A function that represents the depth of the P2 in terms of the y-coordinate and depth of P1. ϕ_{00}, ϕ_{10}, and ϕ_{20} are the parameters of this function obtained by linear regression.

FUNCTION_2:

P3.depth = f (P1.y, P1.depth) or,
P3.depth = $\phi_{01} + \phi_{11}$ * P1.y + ϕ_{21} * P1.depth
--- A function that represents the depth of the P3 in terms of the y-coordinate and depth of P1. ϕ_{01}, ϕ_{11}, and ϕ_{21} are the parameters of this function obtained by linear regression.

2. **FUNCTION_3:**

P3.y = f (P1.x, P1.y, P1.depth) or,
P3.y = $\phi_{02} + \phi_{12}$ * P1.x + ϕ_{22} * P1.y + ϕ_{32} * P1.depth
--- A function that represents the y-coordinate of the P3 in terms of the x-coordinate, y-coordinate, and depth of P1. ϕ_{02}, ϕ_{12}, ϕ_{22}, and ϕ_{32} are the parameters of this function obtained by linear regression.

3. **FUNCTION_4:**

Average depth of all the points between P2 and P3 is represented by AvD_P2_P3.
AvD_P2_P3 = f (P2.x, P2.y, P2.depth) or,
AvD_P2_P3 = $\phi_{03} + \phi_{13}$ * P2.x + ϕ_{23} * P2.y + ϕ_{33} * P1.depth
--- A function that represents the average depth of all the points between P2 and P3 in terms of the x-coordinate, y-coordinate, and depth of P2. ϕ_{03}, ϕ_{13}, ϕ_{23}, and ϕ_{33} are the parameters of this function obtained by linear regression.

4. **FUNCTION_5:**

Slope of the line joining the points P1 and P3 is referred to as Slope_P1_P3.
Slope_P1_P3 = **(P1.depth – P3.depth) / (P1.y – P3.y)**

5. **FUNCTION_6:**

Slope of the line joining the points P2 and P4 is referred to as Slope_P2_P4.
Slope_P2_P4 = (P2.depth – P4.depth) / (P2.y – P4.y)
This parameterized model is created from a set of 59 examples of BGR and depth images of the actual REAL and MODEL up-stairs.

4.5 How the algorithm works:

The interest region is first extracted from every frame of the BGR and depth video feed of the Kinect. This region is then scanned to search for feature points. If there are at least two local minima and two local maxima points along these scans, then (assuming them to be potential stair edges) the points are extracted as the four feature points (P1, P2, P3, P4). As described in the previous sections, the x and y coordinates and the depths of these points are saved for further analysis. Their values are then plugged into the functions of the parameterized model. Now, the algorithm already knows what the output values of these functions should be if the camera is really looking at the Model "up-stairs" case. If we observe that the outputs of the functions are within some close acceptable thresholds of those values, then the algorithm declares that the Model "up-stairs" case is in front of the camera. If there is some other object that the camera is looking at, the functions of the parameterized model will never give proper values all at the same time. This is how the program identifies the up-stairs. Once a stair is found, the edges are marked, and the distance of the edges from the camera is displayed in the final display window, as shown in Fig 4.5.

The values of the parameters of this model and the accepted thresholds for the function outputs are given in the following table.

Function	Parameters: Model "Up-stairs"				Function Upper Threshold	Function Lower Threshold
FUNCTION_1	ϕ_{00} = -97.3592	ϕ_{10} = 0.0585	ϕ_{20} = 0.9768		30	-20
FUNCTION_2	ϕ_{01} = -36.7519	ϕ_{11} = 0.0494	ϕ_{21} = 1.0065		20	-20
FUNCTION_3	ϕ_{02} = -349.7489	ϕ_{12} = -0.0112	ϕ_{22} = 1.0507	ϕ_{32} = 0.2898	30	-20
FUNCTION_4	ϕ_{03} = 28.0292	ϕ_{13} = -0.0081	ϕ_{23} = -0.0013	ϕ_{33} = 1.0301	20	-30

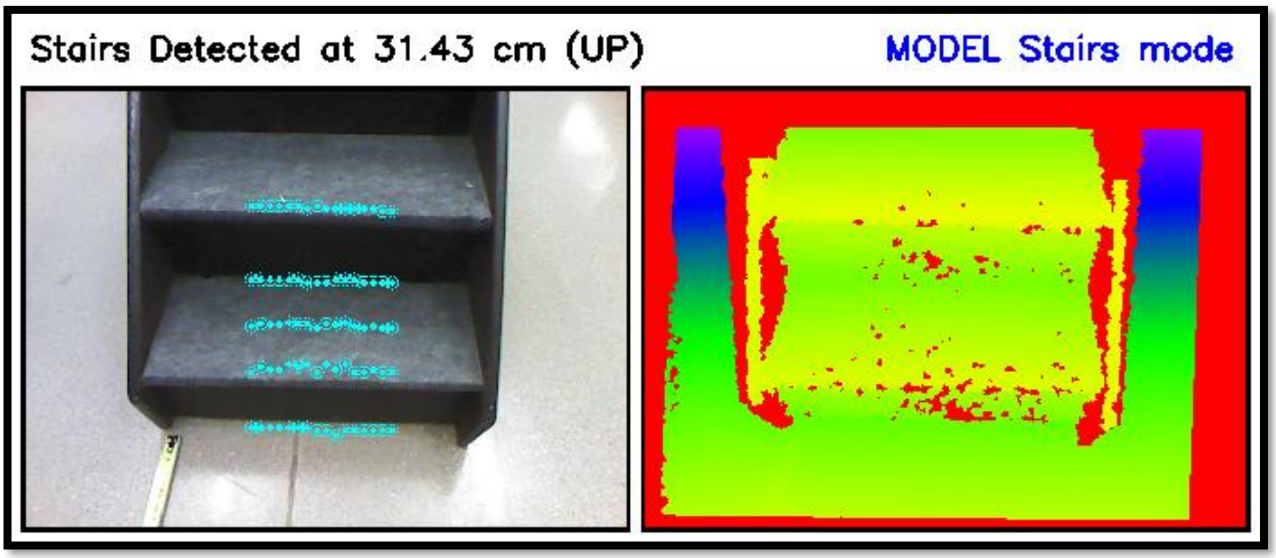


Fig 4.5: Final BGR and Depth images, showing the MODEL up-stairs marked by cyan dots and displaying the distance of the stairs from the viewer.

4.6 Processing of the images for REAL up-stairs:

The algorithm used to detect REAL up-stairs is very similar to the one used for finding MODEL up-stairs. The only difference is in the values of the parameters used. These parameter values for the REAL stairs are given in the following tables.

Parameters for preprocessing: REAL "Up-stairs"	Values
Minimum depth (or distance) below which stairs will not be detected	400 mm
Maximum depth (or distance) above which stairs will not be detected	1400 mm
Angle at which the Kinect is mounted (with the horizontal)	45◦
Height from the ground at which the Kinect is mounted	880 mm
Width of the interest region for down-stairs	320 pixels
Height of the interest region for down-stairs	480 pixels

Function	Parameters: REAL "Up-stairs"				Function Upper Threshold	Function Lower Threshold
FUNCTION_1	ϕ_{00} = -220.2735	ϕ_{10} = 0.0945	ϕ_{20} = 1.0829		60	-20
FUNCTION_2	ϕ_{01} = -5.9768	ϕ_{11} = 0.0802	ϕ_{21} = 1.0612		110	-20
FUNCTION_3	ϕ_{02} = -351.6235	ϕ_{12} = 0.0031	ϕ_{22} = 0.9932	ϕ_{32} = 0.1894	30	-20
FUNCTION_4	ϕ_{03} = 118.7745	ϕ_{13} = -0.0120	ϕ_{23} = 0.0253	ϕ_{33} = 1.0189	100	-30

The following figure shows the detection of REAL up-stairs.

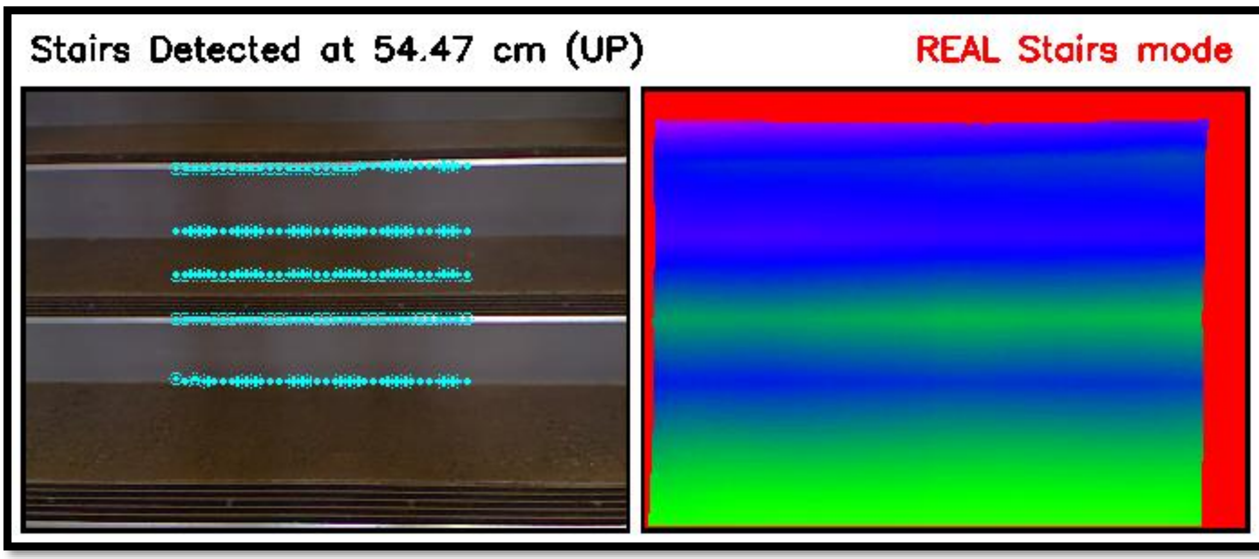


Fig 4.6: Final BGR and Depth images showing the detection of REAL up-stairs.

5. Appendix:

5.1 List of components and their web-links:

The following are the list of components used in the setup hardware along with their respective web-links.

Hardware Components			
Part Name	**Description**	**Qty.**	**Web-link**
Wheel	Uxcell Shopping Wheel Trolley Brake Swivel Caster, 1.5-Inch, Black, 4-Piece	4	https://www.amazon.com/Uxcell-Shopping-Trolley-1-5-Inch-4-Piece/dp/B00PZX2AC8/ref=sr_1_5?ie=UTF8&qid=1483640900&sr=8-5&keywords=swivel+casters
Hex-nut for the wheel	Zinc-Plated Steel Hex Nut, Medium-Strength, Class 8, M8 x 1.25 mm Thread, Pack of 100	4	https://www.mcmaster.com/#90591a161/=160j8q7
Acrylic Sheet	Optically Colored Cast Acrylic Sheet, 1/4" Thick, 12" x 12" – GRAY (For Parts: Part 6)	1	https://www.mcmaster.com/#8505k734/=160j8wk

Acrylic Sheet	Optically Colored Cast Acrylic Sheet, 1/4" Thick, 24" x 36", Gray (For Parts: Part 1 to Part 5, Base, Circuit Base)	1	https://www.mcmaster.com/#8505k738/=160j92k
Corner Bracket	Galvanized Steel Corner Bracket with 1" Long Sides	18	https://www.mcmaster.com/#1556a41/=160j972
20 mm M3 Screw	Black-Oxide Alloy Steel Socket Head Screw M3 x 0.5 mm Thread, 20 mm Long	36	https://www.mcmaster.com/#91290A123
30 mm M3 Screw	Black-Oxide Alloy Steel Socket Head Screw M3 x 0.5 mm Thread, 30 mm Long, Partially Threaded	12	https://www.mcmaster.com/#91290A130
M3 Nut	High-Strength Steel Hex Nut	48	https://www.mcmaster.com/#92497A200

	Class 10, Zinc Yellow-Chromate Plated, M3 x 0.5 mm Thread		
Circuit Base Spacers	Nylon Unthreaded Spacers 3/8" OD, 9/16" Length, for Number 10 Screw Size	4	https://www.mcmaster.com/#94639a847/=163emfu

5.2 List of optional components and their web-links:

The following table shows a list of components (and their web-links), that can be used in the hardware setup or can be optionally excluded as well. This includes the components that are needed for running the setup using a battery. In absence of these components, the Kinect can be powered using its own power adapter from an AC power outlet.

Optional Components			
Part Name	**Description**	**Qty.**	**Web-link**
Battery	Turnigy nano-tech 5000mah 4S 25~50C Lipo Pack	1	https://hobbyking.com/en_us/turnigy-5000mah-4s1p-14-8v-20c-hardcase-pack.html
Voltage Regulator	Power Distribution Board for w/ 12V & 5V BEC	1	http://www.robotshop.com/en/power-distribution-board-quadcopter-12v-5v-bec.html
Switch	Rocker Switch: 3-Pin, SPDT, 10A	1	https://www.pololu.com/product/1406
XT60 Battery Connectors	XT60 Connectors - Male/Female Pair	1	https://www.sparkfun.com/products/10474
JST 2 Pin connector (color: Red)	JST 2 Pin Connector Plug Male & Female Pair	1	https://www.amazon.com/Pairs-Connector-Battery-Discharge-Female/dp/B01JUDP5NY/ref=sr_1_1?ie=UTF8&qid=1485534786&sr=8-1&keywords=jst+power+connector